Quantifying
Uncertainty
First Edition
Josh Mattes

Quantifying Uncertainty

A Course in Statistics Using R and Stan
with Applications in Environmental Science

Version 1.9

Dr. Joshua J. Mattes
Sitting Bull College

Contents

Contents **2**

0 Preface **7**

Measurements ☐ Regression ☐ What Does This Book Cover? ☐ Who Is This Book For? ☐ Why Did I Write This Book? ☐ Acknowledgments and Feedback ☐ Book License

Part I Descriptive Statistics

1 Data Basics **15**

Observations and Measurements ☐ Data Typing ☐ Visually Presenting Data ☐ Labeling Data and Σ–Notation ☐ Exercises

2 A Typical Datum: Averages **25**

The Mode ☐ The Median ☐ The Mean ☐ So Which One Is The "Typical" Datum? ☐ Exercises

3 Deviations From The Mean **31**

An Actual Conversation. Really. ☐ The Variance s^2 ☐ The Standard Deviation ☐ Exercises

4 Quantiles and Box Plots **37**

Definition ☐ Computing Quantiles in R ☐ Interquartile Range ☐ Box Plots ☐ Exercises

5 More About Vectors in R **43**

Data Selection in Vectors ☐ Transforming Data: Changing Units ☐ Factors and Categorical Data ☐ Missing Values and `NA` ☐ Exercises

6 Data Frames **57**

Structure of A Data Frame ☐ Data Selection in Data Frames ☐ Creating a Data Frame ☐ Importing & Exporting Data ☐ Sorting ☐ Exercises

7 More About Data Frames **67**

Making Use of Factor Encoding in Graphs ☐ The `with()` Function ☐ The `pairs()` Function ☐ The `aggregate()` Function ☐ Exercises

8 Limitations of Descriptive Statistics **75**

Measurement Noise ☐ Uncertainties in Regression ☐ Description vs. Inference ☐ Exercises

Part II Probability

9 Introduction to Probability 85
Ranking Plausibility ☐ Probability = Quantified Plausibility ☐ Probability Depends On What You Know ☐ The Complement Rule ☐ The Principle of Insufficient Reason ☐ Exercises

10 The Product and Sum Rules 91
The Product Rule (and) ☐ Independent Events ☐ The Sum Rule (or) ☐ Exercises

11 Bayes' Theorem 99
Statement of Theorem ☐ Meaning and Interpretation of Theorem ☐ Extended Example 1: The Impact of Priors ☐ Extended Example 2: Bayesian Updating ☐ Exercises

12 Binomial Probabilities 107
Computing A Binomial Probability ☐ Efficient Counting ☐ Binomial Experiments ☐ Exercises

13 Discrete Probability Distributions 113
The Binomial Distribution ☐ The `dbinom()` Function in R ☐ Any Single Outcome Is Unlikely ☐ Cumulative Probability and `pbinom()` ☐ The Poisson Distribution ☐ Exercises

14 Probability Densities 123
A Simplified Model of Measurement Noise ☐ From Discrete to Continuous ☐ Probability = Area Under Curve ☐ The Normal Density ☐ Some Other Densities We'll Use ☐ Exercises

Part III Bayesian Inference

15 Bayesian Inference From Binomial Data 137
Formulating the Question & Answers ☐ Organizing Data ☐ Pick Statistical Models ☐ Compute Posterior Probabilities ☐ Credible Intervals ☐ Posterior Density Functions ☐ Exercises

16 Computing Posteriors With RStan 147
Specify Data In R Script ☐ Write Stan File ☐ Call Stan From R ☐ Draws From a Distribution or Density ☐ Compute and Plot Posterior Density ☐ Computing Credible Intervals ☐ Point Estimates ☐ The Cumulative Posterior Density ☐ Informative Priors ☐ Exercises ☐ Appendix - Embedding Stan Model

17 Bayesian Group Comparisons 157
Comparing Groups of Binomial Data ☐ Compute Posteriors for A and B ☐ Compute Credible Intervals ☐ The Posterior for `dp = pA - pB` ☐ So What's Our Answer? ☐ Exercises

18 Bayesian Inference From Normal Data 167
Formulating A Bayesian Analysis ☐ Compute Posterior Draws ☐ Joint and Marginal Posterior Densities ☐ Posterior Density For μ ☐ The Asymmetry of The σ Posterior ☐ The Standard Deviation of Sigma? ☐ Sensitivity Analysis ☐ Exercises ☐ Appendix - Computational Details

19 Comparing More Than Two Groups 177
Data & Models ☐ `rstan` Implementation ☐ Lists and `lapply()` ☐ Point Estimates ☐ Violin Plots ☐ Probability and Interval Calculations ☐ So Which Fertilizer Is Most Effective? ☐ So Which Fertilizer Should We Use? ☐ Exercises

Part IV Frequentist Methods

20 Distribution of Sample Means 189
What Does "Frequentist" Mean? ☐ Samples and Populations ☐ Sample Mean Distribution ☐ Sample Means for Normal Data ☐ The Distribution of Sample Variances ☐ t Distribution of Sample Means ☐ SMD For Non-Normal Data ☐ The Standard Error of the Mean ☐ Exercises

21 Confidence Intervals 199
95% FCI With Known σ ☐ Interpretation ☐ Formally Constructing the FCI ☐ FCIs for μ With Unknown σ ☐ FCI for σ ☐ Group Comparisons Are Problematic ☐ Exercises

22 NHST I - Basic Methodology 209
Motivating The Logical Framework ☐ Single–Sample t tests ☐ Two-Sample t Tests ☐ F-Tests For Group Standard Deviations ☐ Exercises

23 NHST II - Pitfalls 219
Issues With Interpreting p Values ☐ Issues With The General NHST Procedure ☐ The p Value Lottery ☐ p Value Fishing ☐ Exercises

24 NHST III: Power Analysis 229
Classification of Inference Errors ☐ The Effect Size and Cohen's d ☐ Definition of Statistical Power ☐ Compute Power With R ☐ Determining Required Sample Sizes ☐ Exercises

25 NHST IV: ANOVA 239
Comparing Groups ☐ Principle Behind ANOVA ☐ Doing It Simply With R ☐ Statistical Power ☐ Comparison With Bayesian Analysis ☐ Other Types of ANOVA ☐ Exercises

Part V Regression Analysis

26 Linear Regression With One Predictor 249
Framework of a Regression Analysis ☐ Compute Regression Parameters ☐ Evaluating the Model ☐ Visualize Uncertainties in Regression Model ☐ Coefficient of Determination (R-squared) ☐ Least-Squares Regression ☐ Exercises

27 Linear Regression With Multiple Predictors 263
Identify Predictors For `stackloss` Dataset ☐ Compute Parameter Values in R ☐ The Coefficient of Determination ☐ Visualizations of The Model ☐ Prediction Intervals ☐ Exercises ☐ Appendix - Categorical Group Comparisons Via Linear Regression

28 Nonlinearity in Predictors 279
The "Linear" in Linear Regression ☐ Polynomial Regression ☐ The Peril of Overfitting ☐ Logarithmic Transformations ☐ Interaction Terms in Multilinear Regression ☐ Exercises

29 Generalized Linear Models **289**
Logistic Regression □ Generalized Linear Models □ Exercises

Bibliography **297**

Index **299**

CHAPTER 0

Preface

Two tasks that come up a lot in science are the *measurement* of quantities, and figuring out how different quantities relate to each other (known as *regression*). For example, you might be interested in determining how safe it is to drink water from the local river by measuring the amount of E. coli bacteria in the water. Furthermore, you might want to determine if the bacterial counts are related to things like the local population density and regulations on local business. What does statistics have to do with measurement and regression?

0.1 Measurements

Let's start with the need for statistics when making measurements. Measuring the E. coli is something scientists know how to do. Collect some water, somehow incubate the growth of the bacteria in the sample until it is easily countable, and then work backwards to see what must have been there when you first collected the sample. You end up with some number, say 75 bacterial colonies per 100 mL of water. So you might claim that, on average, the local river has 75 colonies per 100 mL. So where does statistics come in? It's simply that there is a lot of *uncertainty* in this single estimate of 75 colonies. What are some of the reasons for this uncertainty?

This book will have many margin notes like this one. In addition to containing figures and tables, they will also contain points of clarification, or comments that you may hold off reading until you've had a first pass through the section.

1. The amount of bacteria in the water likely changes from location to location. Some spots in the water might have a faster moving current, some might have more sunlight, some might be more stagnant, etc. Depending upon where we collect water samples in the river, we might have ended up with more or less than 75 colonies per 100 mL.

2. The amount of bacteria in the water likely changes with time. Some days may have more pollution than others, depending upon how hot it is during the day, or on whether a bunch of pollutants just got dumped into the river, or on whether there was a heavy rain and the sewers overflowed, etc.

3. Even at a precise "location" in the water, the amount of bacteria in our water sample will vary. On *average* there may be a true number of bacteria per 100 mL, but in some samples I'll get slightly more than this, and in others slightly less, just because of where the individual bacteria happen to be at the moment. This is no different than flipping a fair coin 10 times. Sometimes you'll get 4 heads, other times 6, and so on.

4. We didn't actually directly count the bacteria. We incubated them at some temperature under certain conditions for a certain amount of time, and we're using mathematical equations to predict exactly how much the bacterial numbers get amplified, so that we can "work backwards" and infer how much was

> there at the start. However, different groups of bacteria will grow at different rates (because of genetic differences, for example). Furthermore, how much growth we expect depends sensitively on the exact incubation temperature, and this temperature will vary a little bit even in a really nice piece of equipment. And so on. In other words, the measurement process itself is subject to uncertainty. When a piece of equipment gives you a measurement, it isn't an exact number delivered by God, but rather an approximate number with inherent uncertainty.

For reasons like this, it would be ridiculous to make a blanket claim such as

> "The amount of E. coli in the river is 75 colonies per 100 mL."

The only sensible and honest thing to do is to admit that there is a good amount of uncertainty in the answer. But how do we quantify this uncertainty? That's where probability theory and statistics come in. Ultimately the goal is to make a statement that's more like

> "Given the various assumptions we've made, there is a 95% chance that the true average value of the E. coli count in the faster-flowing regions of the river is somewhere between 61 and 89 colonies per 100 mL. If there has been recent flooding, instead there is a 95% chance that the count is between 150 and 325."

Note that this statement isn't perfect, because there's still lots of stuff we're leaving out of the story, but it's a reasonable summary. The point is that we're acknowledging the uncertainty in our measurements and estimates. Note that instead of giving an exact number, known as a *point-estimate*, we're giving a range or *interval* of possible values for the bacterial count. We'll learn how to do this in Parts III and IV of the book.

0.2 Regression

As we just discussed, properly interpreting measurements isn't trivial. Performing *regression analysis*, i.e. deducing the relationships between various measurable quantities in the world, is typically a *much* tougher problem. We'd like to understand how the expected bacterial count in the water depends upon other measurable factors, such as:

- water speed
- water temperature
- time of year
- soil composition
- general climate
- populations of other organisms in the water

- patterns of rainfall
- details of fertilizer use on local farms
- composition of effluents dumped into river from local businesses
- local population of human beings
- the type of waste disposal used by local inhabitants

and so on. Because there are so many different factors that could influence the bacterial levels, it's a very challenging problem to figure out exactly which factors are important. Typically you need to ignore most of the factors, and just study a few. For example, we might decide to study the impact of water temperature T and local human population P on the bacterial counts B. Two difficulties arise immediately:

1. The precise mathematical form of the relationship is usually not known ahead of time. Is the relationship additive, i.e. $B = aT + bP$ where a, b are constants? Is the relationship multiplicative, i.e. $B = aPT$, where a is a constant? Is it something else entirely? It's often really hard to know ahead of time, and furthermore it's often really hard to see the proper relationship in the data itself because ...
2. We're ignoring all the other factors, and thus we already know that B won't be predicted solely on the basis of P and T. As such, the data will have a lot of `noise`. By "noise" we mean variation in the bacterial counts B that isn't explained solely in terms of P and T. This noise introduces lots of uncertainty into our analysis, making it much tougher to deduce the precise relationship between P, T, and B.

In this book, particularly important terms and equations are `highlighted`.

As a result, there are two huge sources of uncertainty in regression. One is that the constants a, b etc. themselves have a lot of uncertainty, simply because the measurements (i.e. data) used to determine them are also uncertain. Secondly, it's not usually certain that we've picked the correct mathematical relationship, or *mathematical model*. This is known as *model uncertainty*, and is much tougher to quantify. Again, the only logical and honest way to proceed is acknowledge our uncertainty, and use statistics to quantify it as best as possible.

0.3 What Does This Book Cover?

Statistics is a huge subject, and there's an unfathomably large amount of material that could be studied. However, the aims of this book are modest. The two primary topics discussed in this book are the quantification of uncertainty in measurements, and the quantification of uncertainty in the parameters (i.e. constants) of a regression equation. Quantifying model uncertainty is extremely important, but is a bit beyond the scope of this book.

Parts I and II provide some necessary background material. Parts III offers an introduction to quantifying uncertainty in the framework of Bayesian statistics. Bayesian methods are logical, intuitive, and a wonderful tool for quantifying uncertainty in the age of modern computers. However, the majority of scientists use

the framework of Frequentist statistics, and it's important to be able to interface with other people/reports/etc. Hence, Part IV of the book provides a brief introduction to some popular tools in frequentist statistics. In Part V, with the experience of the first four parts under our belts, we discuss regression analysis.

0.4 Who Is This Book For?

The intended reader is somebody who

1. has access to a modern computer, and can install the freely-downloadable software R and STAN
2. has taken algebra, and remembers a little bit of it
3. is willing to put up with the occasional scary-looking equation, assuming that all its pieces are explained
4. wants an introduction to quantifying uncertainty of measurements and regression parameters primarily using Bayesian methods
5. is okay with letting the computer provide good approximations, rather than working out all the formulas exactly by hand when possible
6. has a leaning towards the biological/environmental sciences (simply because most of the examples are from this area; the ideas are of course general, and could apply to any type of data)

0.5 Why Did I Write This Book?

When tasked with teaching a statistics course for environmental science students, I searched for a textbook meeting the following criteria:

1. didn't invoke calculus or complicated algebra
2. used Bayesian methods
3. utilized free software (e.g. R)
4. had examples drawn from environmental science and biology
5. was geared towards upper-level undergraduate or entry-level graduate students

McElreath's *Statistical Rethinking* [1] is a marvelous book, but in my estimation is too advanced for many undergraduates and entry-level grad students.

I was unable to find a text that met these criteria. As such, I initially picked a textbook that didn't quite fit the ccourse, and was continually supplementing it with notes I had typed up. Eventually it made sense to expand and transform these notes into the primary course text, resulting in this book. Hopefully you find it useful in some capacity.

0.6 Acknowledgments and Feedback

Many students and colleagues helped to improve this book. Some engaged in lengthy conversations about the content and focus of the book. Some provided datasets used in the text. Others carefully read early drafts and offered numerous suggestions and corrections. In particular, I want to thank Saul Bobtail Bear, Kimberly Blevins, Samantha Borah, Dan Buresh, Wes Craddock, Bruz Van Dusen, Tim Krahler, Maurianna Loretto, Mafany Mongoh, and Floris White Bull. Enormous thanks also goes to my wife Sara, who put up with many weekends of me being glued to my computer all day.

Any errors, inaccuracies, or other flaws in the book are of course the sole responsibility of the author. Comments, suggestions, and corrections are always welcome, and can be sent to mattes.josh@gmail.com.

0.7 Book License

This book is published under a Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) License. This means you can freely share, copy, adapt, and transform the material of this book as long as you give credit to the author and re-use the same Creative Commons License. See `https://creativecommons.org/licenses/by-sa/4.0/` for more details.

Part I

Descriptive Statistics

CHAPTER 1

Data Basics

`Descriptive statistics` are numbers or visuals that merely describe data, without any assumptions or inferences.

Methods involving the *inference* of things that go *beyond* the data are known as `inferential statistics`. By their nature, these methods lead to conclusions that are imperfect, probabilistic, and debatable on some level.

If you had all possible information, descriptive statistics would tell you everything there is to know, and you likely wouldn't have any need to infer anything, i.e. to use *inferential methods*. On the other hand, if you're missing information, i.e. you don't have all the facts, then clearly your conclusions can't be certain.

This chapter introduces the concepts of data and descriptive statistics. The basic ideas of observation and measurement are introduced in Section 2. Various ways of displaying data are then discussed in the following sections.

1.1 Observations and Measurements

A `datum` (plural is *data*) is a fact or observation or measurement related to a situation. It's basically a synonym for "piece of information", with the caveat that data is more something that can be directly measured and observed, and less the result of a complex judgment of some sort. That being said, there are no sharp criteria that allow a piece of information to be classified as a datum.

Nobody would argue that the weight of an object is a datum. On the other hand, complex judgments such as "dogs are the best kind of pet" or "he is smart" are in most cases not considered to be data, because of how vague and subjective they are. Unless you're collecting information on people's *opinions*, in which case such statements could be construed as data. Context is crucial.

Note that you always have choices to make when recording data. For example, say you're trying to collect data on the weight of members of an animal population. You could weigh the animals to the nearest tenth of a kilogram, in which case your data are real numbers (e.g. the weight of one animal is 34.7 kg). You could also simply categorize the animals as small (0 to 25 kg), medium (25 to 50 kg), or large (50 and above) so that your data are simple letters or categories (S, M, or L).

We again emphasize that there are *always* choices to make when collecting data. For example, one may think that recording sex data is necessarily a matter of writing Male or Female, but this is merely a biological approximation. There is a highly complex multidimensional genetic and phenotypic spectrum of sexual characteristics even in simple organisms. Reducing this complexity to binary categories is simply a choice on the part of the person collecting data, and not an inherent feature of reality.

The same *feature* of reality (in this case an animal weight) can thus be recorded in different ways, as different *types* of information (integers, real numbers, categories, etc.) depending on how you choose to collect, classify, and record the data. These different *data types* may involve different statistical methods, so it is useful to establish a common language and concept of data types.

1.2 Data Typing

There are two fundamental types of data: `quantitative` (numerical) data and `qualitative` (non-numerical) data, each with a few sub-classifications. These are summarized in Table 1.1 below.

When qualitative data comes from a finite number of possibilities, which is often the case, then such data is sometimes referred to as `categorical data`.

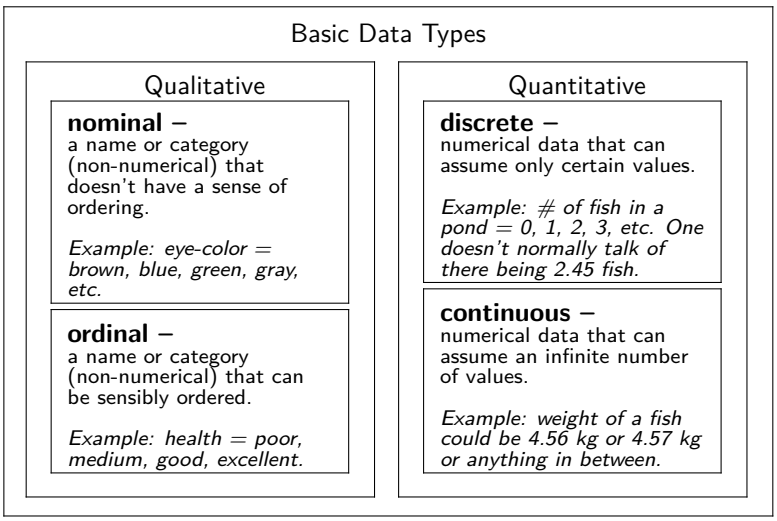

Basic Data Types
Qualitative
nominal –
a name or category (non-numerical) that doesn't have a sense of ordering.
Example: eye-color = brown, blue, green, gray, etc.
ordinal –
a name or category (non-numerical) that can be sensibly ordered.
Example: health = poor, medium, good, excellent.
Quantitative
discrete –
numerical data that can assume only certain values.
Example: # of fish in a pond = 0, 1, 2, 3, etc. One doesn't normally talk of there being 2.45 fish.
continuous –
numerical data that can assume an infinite number of values.
Example: weight of a fish could be 4.56 kg or 4.57 kg or anything in between.

Table 1.1: Basic data type classifications.

Again we note that features of nature don't really have predefined data types; rather, data types reflect the manner in which you conceptualize, measure, and record data. For example, consider measuring eye-color. If you measured this as "green, brown, blue" etc., then the data is qualitative and nominal. However, if you measure the exact spectrum of light-frequencies, then the color data becomes quantitative and continuous. Another example: you could measure ages in a population and record the data as "adult" or "juvenile"; the measurement is then qualitative, ordinal, and dichotomous. On the other hand, you could record actual ages in days, in which case the data is quantitative and continuous.

`Dichotomous data` or *binary data* just means that there are only two possible values of a datum, such as 1/0, male/female, yes/no, etc.

1.3 Visually Presenting Data

Before we can analyze or assess data, we usually need to put it into a more comprehensible form, often a table or chart of some sort. For the next few sections we will consider the example of measuring the body temperatures of twenty-five crabs placed in 24.3°C air.[1] You record your data (temperatures in °C) in your notebook as follows:

25.8, 24.6, 26.1, 22.9, 25.1, 27.3, 24.0, 24.5, 23.9, 26.2, 24.3, 24.6, 23.3,
25.5, 28.1, 24.8, 23.5, 26.3, 25.4, 25.5, 23.9, 27.0, 24.8, 22.9, 25.4.

Even though the measurements indicated look like single numbers, they *are not* single numbers. *Measurements of continuous numerical quantities are always ranges of numbers.* Presumably, the thermometer you were using was only accurate to the nearest 0.1°C. Your first measurement of 25.8°C therefore does not mean the temperature of the crab was 25.80000000000000000...°C. Rather, you are indicating that the crab is somewhere between 25.75°C and 25.85°C.

1.3.1 Sorting and Listing in Tables

At this point, if you haven't yet done so, please install R and an environment like RStudio, work briefly through one of the numerous free R tutorials online, and make sure you know how to run an R-script.

Unless the data is *paired* with other measurements (not the case here), it makes sense to sort the data. Let's see how to do this easily in R. First we store the data as a group (we choose here to call it `x`) by issuing the command

[1] Data was taken from example problems in Zar's *Biostatistical Analysis*, 4th Ed. [2]

(R.1.1)
```
x <- c(25.8, 24.6, 26.1, 22.9, 25.1, 27.3, 24.0, 24.5, 23.9,
       26.2, 24.3, 24.6, 23.3, 25.5, 28.1, 24.8, 23.5, 26.3,
       25.4, 25.5, 23.9, 27.0, 24.8, 22.9, 25.4)
```

As we learn about R in this book, often we will color-code commands.
Blue = built-in function.
Pink = user-defined variable.
Purple = adjustable parameter in function.
Black = numbers or text that should be interpreted literally by R.

Note that when you type in code in R, the colors probably won't look like this.

The `<-` operator assigns what's on the right to whatever variable name you specify on the left. The `c()` function collects all of the comma-separated numbers into what's called a `vector`, so that R knows these numbers all belong together. You can then type `sort(x)` to display the data in ascending order. The sorted data looks like

22.9, 22.9, 23.3, 23.5, 23.9, 23.9, 24.0, 24.3, 24.5, 24.6, 24.6, 24.8, 24.8,
25.1, 25.4, 25.4, 25.5, 25.5, 25.8, 26.1, 26.2, 26.3, 27.0, 27.3, 28.1

If we want a nice display of the data in a presentation or report, we could also display the numbers in a vertical table, as shown in Table 1.2. Horizontal lines can be used to separate the entries, if it makes reading the table easier. A horizontal line of info in a table is known as a `row`, and a vertical line of info in a table is known as a `column`.

crab #	temp(°C)
1	22.9
2	22.9
3	23.3
4	23.5
5	23.9
6	23.9
7	24.0
8	24.3
⋮	⋮
23	27.0
24	27.3
25	28.1

Table 1.2: Crab temperature data sorted in ascending order. The vertical dots after crab #8 indicate that not all of the data are shown, usually in the interest of saving space on the page. Note that the crab # here does not refer to the original order in which the data were recorded, due to the sorting.

Even though the numbers are sorted and presented in a nice table, it's still a lot of information to process. It's tough to simultaneously visually process 25 3-digit numbers. A routine solution to this difficulty is to group the data into categories of similar value, which we now discuss.

1.3.2 Binning Data

We could put all temperatures $22.0 < T \leq 23.0$ into one group, all the temperatures $23.0 < T \leq 24.0$ into another group, and so on. These groups are called `bins`, and grouping the data into bins is called `binning`. Looking at Table 1.2, we see that crabs #1–2 fit into our first bin, crabs #3–7 fit into our second bin, and so on. Table 1.3 gives the number of temperatures in each bin. This number of data points per bin is called the `frequency`.

temp (°C)	frequency
(22,23]	2
(23,24]	5
(24,25]	6
(25,26]	6
(26,27]	4
(27,28]	1
(28,29]	1
(28,29]	1

Table 1.3: Crab temperature data sorted into seven bins.

After binning, there are only 7 table rows to examine, as opposed to 25. Table 1.3 makes it a bit easier to see that the majority of the temperatures are clustered around 25°C, and the numbers dwindle as you move away from this temperature.

The size and location of the bins are somewhat arbitrary, though you don't want them too narrow (nothing in most bins) or too wide (everything in a few large bins). Finally, note that Table 1.2 contains more information than Table 1.3. It is often the case that one loses information when creating simplified visuals of the data.

1.3.3 Histograms

In addition to listing the binned data in a table, one can also make a bar-chart of the bins. This type of chart is known as a `histogram`. To generate a histogram of our data-set `x` in R, issue the command `hist(x)`. The resulting histogram is shown in Fig. 1.1. We can put labels on the histogram by adding some optional parameters to the `hist()` function:

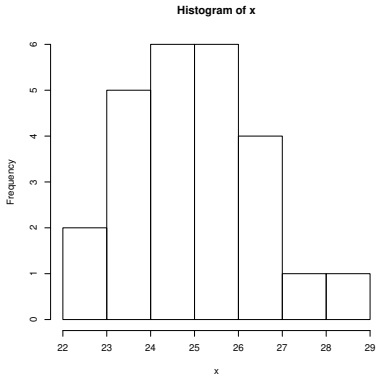


Figure 1.1: Histogram of crab temperature data produced by R.

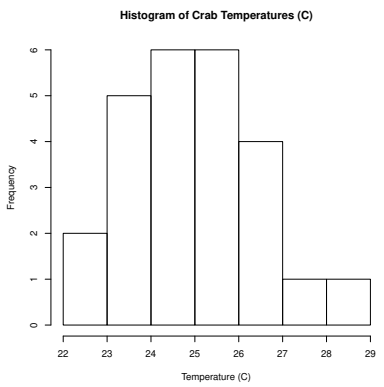


Figure 1.2: Better labeled histogram of crab-temperature data.

```
hist(x, main="Histogram of Crab Temperatures (C)",
     xlab="Temperature (C)")
```
(R.1.2)

with results shown in Fig. 1.2. The `hist()` function has many options that can be adjusted, including the bin width/location/number, the color and shading of the bars and axes, etc. Consult the internet for more info.

1.3.4 Paired Data

If is often the case that a datum is naturally paired with another datum. For example, one could measure the height of a plant at different times, as shown in Table 1.4. Clearly each height is naturally paired with the corresponding time, in what is known as a `time series`.

time (days)	height (cm)
0.0	0.0
1.1	1.4
2.3	1.9
2.9	4.6
4.3	5.9
5.0	7.3
6.2	9.2

Table 1.4: Plant height vs. time.

For paired data such as this, histograms don't tend to work well. Instead, it is more usual to plot the paired data as (x, y) coordinate pairs. We issue the following code in R to generate this plot:

```
time <- c(0.0, 1.1, 2.3, 2.9, 4.3, 5.0, 6.2)
height <- c(0.0, 1.4, 1.9, 4.6, 5.9, 7.3, 9.2)
plot(x=time, y=height, main="Height vs. Time")
```
(R.1.3)

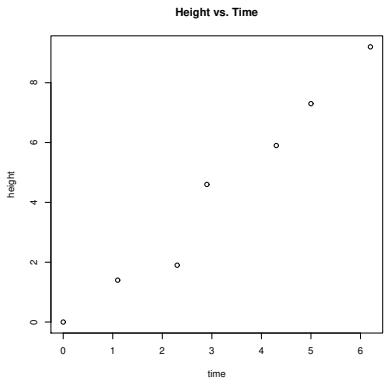


Figure 1.3: Plant height vs. time paired-data plot generated by R.

The result, known as a `scatter plot`, is shown in Fig. 1.3. As before, one can customize the graph in a variety of ways through specifying various plot options. One can overlay lines connecting the points by issuing an additional command using the `lines()` function, with results shown in Fig. 1.4:

```
plot(x=time, y=height, main="Height vs. Time")
lines(x=time, y=height)
```
(R.1.4)

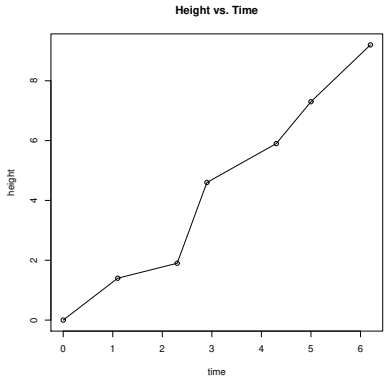


Figure 1.4: Plant height vs. time line-plot generated by R.

Additional types of plots and plot-features will be introduced as the corresponding statistical ideas and techniques are introduced.

1.3.5 Misleading Graphs

A *misleading graph* or *distorted graph* is a graph that misrepresents data. The most familiar example is the frequently-used pie-chart. A pie chart misrepresents data because people tend to be far better at judging ratios of lengths than they are at judging ratios of areas or volumes. This distortion of data through pie-charts is very well known.[2] Despite the ubiquity of pie-charts, and despite the fact that they can be aesthetically pleasing, they don't provide an accurate visual representation of data.

There are many other ways to misrepresent data graphically, including various manipulations of the data and/or the axes. See Wainer's *Visual Revelations* for a nice discussion [3]. For a variety of examples taken from news outlets and government reports, see the *Statistics How-To* website [4].

[2] As a simple illustration of this fact, on 2016 April 16, 8:00pm CST, a Google search of "pie charts are misleading" returned about 340,000 hits.

1.3.6 Organizing Collected Data

A typical format for organizing scientific data is to place it in a table, where each row represents a sample/object of interest, and each column represents a specific type of information being measured/collected for the corresponding sample or object. Such tables are sometimes referred to as `tidy` tables.

For example, say I'm trying to understand how the sex and family-income level of freshman college students is related to their GPA. In this case the sample/object I'm studying is a college student, so that each row of the table represents a different college student. The things I'm measuring for each student are sex, family-income, and GPA, so each of these things would have a column. A sample table for such data is shown in Tab. 1.5.

student	sex	income (k$)	GPA
1	F	25	3.7
2	M	32	2.9
3	M	45	3.4
4	F	67	2.1
5	F	68	2.7
6	M	95	3.7
7	M	112	3.1

Table 1.5: Sample data table for a study of the relationship between sex, income, and GPA of college freshman.

Note that different parts of the table may have different data-types. For example, the sex of a student is a nominal and qualitative datum, whereas family-income and GPA are continuous and quantitative data. Note that the column titled "student" isn't data at all, but rather is just a way of labeling the different students that were studied.

1.4 Labeling Data and Σ–Notation

1.4.1 Subscripts and Summation Notation

Let's introduce a very common notation used in statistics. To motivate this notation, imagine that as part of your data-analysis you need to add up all of the heights shown in Table 1.4. You could grab a computer and compute $0 + 1.4 + 1.9 + 4.6 + 5.9 + 7.3 + 9.2$ and obtain 30.3. In R, one could issue the command `sum(height)` to get this result, where the `height` vector was defined in (R.1.3) in Sec. 1.3.4.

The proper design of experiments requires describing our methods of analysis *before* we collect the data. We could describe our method in words, simply writing down the phrase

"then add up all of the heights you measure."

It's legitimate to do so, but writing out *all* of our mathematical procedures in this way results in a *lot* of words, and often it's hard to read.

Instead of writing out words, we could write an equation, with variables representing the unknown heights that we'll be soon measuring. We learn in algebra to denote unknowns or variables with letters such as A or x. Thus, we could define the first height we'll measure by the letter "A", the second height by the letter "B", and so on. We could then describe our process of adding up all the heights by

$$A + B + C + \cdots + \text{final height}$$

where the three dots ... indicate that one is supposed to continue the obvious pattern (i.e. also add D and E and so on). This notation is again legitimate, but it suffers from a few defects. First, unless you know ahead of time how many heights you'll measure, you don't know which letter to end the formula with, hence the presence of "final height" in the above formula. A second defect is that you quickly

run out of different letters for large data sets. A final defect is that even though all of the numbers you're adding represent the same type of physical thing, in this case a height in cm, the notation doesn't reflect this. It's not clear that $A, B, C, \ldots$ all represent the same kind of quantity.

The simultaneous solution to all of these defects is to use the exact same letter for every measurement, and to distinguish the different measurements with a subscript or *index*. Therefore, we may pick the single letter x to collectively represent all of the measurements, and x_1 refers to the first measurement, x_2 refers to the second measurement, and so on. We could let n denote the number of measurements we'll end up taking, and then write our sum as

Of course we don't need to pick "x". We can pick any letter we like. Perhaps you prefer using a letter like "h" to represent heights, in which case the data are denoted h_1, h_2, etc. Use whatever symbols you feel like, as long as you are clear and consistent.

$$x_1 + x_2 + \cdots + x_n$$

With this notation in place, we can introduce an even more compact way of expressing the idea of **S**ummation. We use the Greek letter **S**, called **S**igma, which is written as Σ, in the following way:

$$\sum_{i=1}^{n} x_i \equiv x_1 + x_2 + \cdots + x_n \tag{1.1}$$

Note that the subscripts are just labels, and don't instruct you to "do anything". This can be confusing at first, because in algebra we learn that the expression $2x$ is an instruction to multiply x by 2, and the expression x^2 is an instruction to multiply x by itself. But the 2 in x_2 isn't any kind of instruction – it's just a label used to distinguish it from the other values of x. It's no different than if we wrote x_{bob} or $x_{\text{second measurement}}$, aside from being conveniently compact.

where the symbol "$\equiv$" means "is defined as". The text below the Σ symbol, $i = 1$, indicates the starting value of the subscript, and the text on top, n, denotes the final value of the subscript.

Expressions written in Sigma-notation, such as Eq. (1.1), are so routine that they are often even further abbreviated as

$$\sum x \quad \text{or} \quad \Sigma x$$

where it is understood that you sum up all the x-values over all of the subscripts. Many statistics books and articles will thus use the very compact expression Σx to indicate things like "add up all of the heights".

1.4.2 Visually Interpreting Σ–Notation

Consider a small data-set where $x_1 = 4, x_2 = 7, x_3 = 11, x_4 = 6$. Suppose we needed to compute $\sum_{i=1}^{4} (x_i - 7)^4$. To visualize this calculation, first note that the index i ranges from 1 to 4. Since the index i will assume the four values 1,2,3,4, we can visually organize the calculation by drawing four boxes, where each box will have a different value for the variable i:

$$\sum_{i=1}^{4} (x_i - 7)^4 = \underset{i=1}{\boxed{(x_i - 7)^4}} + \underset{i=2}{\boxed{(x_i - 7)^4}} + \underset{i=3}{\boxed{(x_i - 7)^4}} + \underset{i=4}{\boxed{(x_i - 7)^4}} \tag{1.2}$$

Now in the first box, we have $i = 1$. This means that in the expression inside this first box we replace the i in x_i by 1 to get $x_1 = 4$. Thus, the first term has the value $(4 - 7)^4 = (-3)^4 = 81$. Continuing with $i = 2$, $i = 3$, and $i = 4$ we obtain

$$\sum_{i=1}^{4}(x_i-7)^4 = \underset{\mathbf{i=1}}{\boxed{(x_{\mathbf{i}}-7)^4}} + \underset{\mathbf{i=2}}{\boxed{(x_{\mathbf{i}}-7)^4}} + \underset{\mathbf{i=3}}{\boxed{(x_{\mathbf{i}}-7)^4}} + \underset{\mathbf{i=4}}{\boxed{(x_{\mathbf{i}}-7)^4}}$$

Data Set:
$x_1 = 4$
$x_2 = 7$
$x_3 = 11$
$x_4 = 6$

$$= \underset{\mathbf{i=1}}{\boxed{(x_{\mathbf{1}}-7)^4}} + \underset{\mathbf{i=2}}{\boxed{(x_{\mathbf{2}}-7)^4}} + \underset{\mathbf{i=3}}{\boxed{(x_{\mathbf{3}}-7)^4}} + \underset{\mathbf{i=4}}{\boxed{(x_{\mathbf{4}}-7)^4}}$$

$$= \underset{i=1}{\boxed{(4-7)^4}} + \underset{i=2}{\boxed{(7-7)^4}} + \underset{i=3}{\boxed{(11-7)^4}} + \underset{i=4}{\boxed{(6-7)^4}}$$

$$= \underset{i=1}{\boxed{(-3)^4}} + \underset{i=2}{\boxed{(0)^4}} + \underset{i=3}{\boxed{(4)^4}} + \underset{i=4}{\boxed{(-1)^4}}$$

$$= 81 + 0 + 256 + 1$$
$$= 338$$

What if instead we were supposed to compute something like $\frac{1}{4}\sum_{i=1}^{4}(x_i-7)^4$? Just compute the sum as before, and multiply the result by 1/4.

1.4.3 Example of Computing A Sum In R

Suppose we needed to compute

$$\frac{1}{n}\sum_{i=1}^{n}(x_i-25.028)^4 \tag{1.3}$$

The quantity 25.028 is actually the *mean* of the data in (R.1.1). We show how to compute this in Sec. 2.3. The sum shown in Eq. (1.3) is actually part of the computation for the *kurtosis* of a sample, which relates to the symmetry of the data.

for the vector `x` of n temperature data defined in (R.1.1). We first create a vector containing the value of each term in the sum:

(R.1.5)
```
terms <- (x-25.028)^4
```

Reminder that `R code in pink` in this book represents names you are free to choose. Hence, for example, `terms` could have been any other name you feel is useful.

Operations such as subtraction and exponentiation are *vectorized* in R. What this means is that in (R.1.5), the number 25.028 is first subtracted from each number in `x` to produce a new vector. Then, each element of this new vector is raised to the fourth power, resulting in the vector `terms`. Then we can sum all the terms together via

(R.1.6)
```
terms.sum <- sum(terms)
```

You are allowed to use periods in user-defined names in R. Something like `terms.sum` is easier to read than `termssum`. Something like `termsSum` is also a bit easier to read. Note that R distinguishes upper-case vs. lower-case letters, so `termsSum` is different than `termssum`.

which we can then divide by $n =$ `length(x)` to obtain the final answer

(R.1.7)
```
answer <- terms.sum/length(x)
```

which has the value of approximately 7.89. Note that you can combine (R.1.5), (R.1.6), and (R.1.7) into a single line as

```
answer <- sum((x-25.028)^4)/length(x)
```
(R.1.8)

but doing so requires careful use of parentheses and is tougher to read.

1.5 Exercises

1. In each of the following data-gathering situations, visualize the raw data as a table with rows for sample subjects and columns for variables by doing the following two things:

   - Complete the statement *"In this data set there would be one row for each _____".*
   - For each column, complete the statement *"In this data set there would be a column for measurements of _____, which is a _____ variable."*, where the last blank indicates all descriptors of the data type (i.e. qualitative, quantitative, nominal, ordinal, dichotomous, discrete, continuous).

   For example, if I were looking at litter size and health (poor, medium, good) in foxes, an appropriate answer is *"In this data set there would be one row for each litter. There would be a column for measurements of litter size, which is a quantitative and discrete variable. There would also be a column for litter health, which is a qualitative and ordinal variable."*

   Here are the data-gathering situations:

   i) I'm studying the life expectancy of Pacific white-sided dolphins in their natural range from the Gulf of Alaska to Baja California.

   ii) I'm examining whether there is a relationship between hair color and sex in humans (for example, is black hair more frequent in males than females?).

   iii) In an experiment I'm planning, 10 Drosophila eggs will be put in a vial and the positions of the pupae will be noted after pupation. Some will pupate at the margin (or the wall) of the vial (these are called peripheral); others will pupate away from the wall (these are called central). I'm interested in the proportion of central pupae.

   iv) By observation of a sample of 12 nesting pairs of Stellers jays, I make a record of the directions (angle clockwise from due North) relative to the tree trunks in which they've chosen to locate their nests (for example, a nest on a branch pointing due East would have a reading of 90°), to see if they tend to favor South-facing nesting sites.

2. The following data are measurements of the inter-orbital width (how far apart the eyes are) in a sample of 40 domestic pigeons. Measurements are in mm.

2.2	12.9	11.8	11.9	11.6	11.1	12.3	12.2	11.8	11.8	10.7
11.5	11.3	11.2	11.6	11.9	14.3	11.2	10.5	11.1	12.1	11.9
10.4	10.7	10.8	11.0	11.9	10.2	10.9	11.6	10.8	11.6	10.4
10.7	12.0	12.4	11.7	11.8	11.3	11.1				

The following (a through d) should be done with pencil and paper (no computers):

i) Bin the measurements and give the results in a table.

ii) Make a histogram based on your binning. If the width of the bars needs adjusting, redo part i) accordingly.

iii) How would you describe the shape of the histogram?

iv) Research what is meant by the statistics term `outlier`, and give a definition. Are any of the inter-orbital width observations outliers? Explain/justify.

Now use R to generate 2 different histograms of the data, each with a different bin width. The number of bins (or correspondingly the bin width) can be adjusted via the `breaks` option in the `hist()` function. Look this up for examples.

To find more info on using `breaks` parameter of the `hist()` function you could do a web search, or use the help functions in R, etc.

3. Pick one specific day of the week. Make paired observations of the time (recorded as hours past midnight) and the average temperature in Bismarck, ND. You should have 25 data points (i.e. your times should be 0 hrs, 1 hr, 2 hrs, ... 24 hrs). You can also use the internet to find this time/temperature data.

   Use R to make a plot of temperature vs time. Add an overlay of lines connecting the data points. Make sure the axes are labeled and that the plot has an informative title.

4. In this exercise we lay some mental ground-work for the important topic of linearly transforming data, which we shall discuss further in Sec. 5.2. Consider a set of 3 data $h_1 = 4$, $h_2 = 3$, $h_3 = 9$. Do the following by hand, though you may use R to check your work.

   i) Compute $A = \sum_{i=1}^{3} h_i$.

   ii) If we now add 5 to each h_i, what is the sum? In other words, what is $B = \sum_{i=1}^{3} (h_i + 5)$?

   iii) What is the simple relationship between B, A, $n = 3$, and the number 5? Your answer should be an equation with B on the left, and an expression involving A, 3, and 5 on the right. What is the explanation for this simple relationship?

   iv) Instead of adding 5 to each h_i, instead suppose we multiply each h_i by 5. What is $C = \sum_{i=1}^{3} 5h_i$?

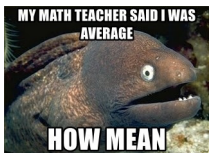
   v) What is the simple relationship between C, A, and 5? What is the explanation for this simple relationship?

5. Use R to compute $\frac{1}{n} \sum_{i=1}^{n} \sqrt{x_i^2 + 3}$ for the vector `x` defined in (R.1.1).

CHAPTER 2

A Typical Datum: Averages

A common question when looking at data is "what is the *typical* datum?" It's surprisingly difficult to come up with a mathematically precise definition of *typical* or *average*. We will discuss the three common definitions of *average*: the mode, median, and mean.



2.1 The Mode

Definition

Consider the data set $x = \{5, 10, 10, 15, 20, 25\}$. What is a "typical" datum? One possible answer is *"well, 10 shows up more than anything else, so the typical datum is 10".*

$$\underbrace{5 \quad 10 \quad 10 \quad 15 \quad 20 \quad 25}$$

The value 10 shows up more often than any other datum value.

The statement $x = \{5, 10, 10, 15, 20, 25\}$ is a shorthand for saying $x_1 = 5$, $x_2 = 10$, $x_3 = 10$, etc. It is the equivalent of the command `x <- c(5,10,10,15,20,25)` in R.

This notion of *typical* or *average* – the datum value with the highest frequency – is called the `mode` of the data.

Notice that the concept of the mode does not rely upon the data being quantitative, and hence works just as well for all types of qualitative data.

Binned Data

The concept of the mode can be extended to binned data by indicating which bin contains the most data points (i.e. the bin with the highest frequency). Referring to Table 1.3, we see that the bins $24°\text{C} \to 25°\text{C}$ and $25°\text{C} \to 26°\text{C}$ have the highest frequency, and thus the mode is the interval of numbers $24 \to 26°\text{C}$. This can also be seen in the histogram of Fig. 2.1 by identifying the tallest bars.

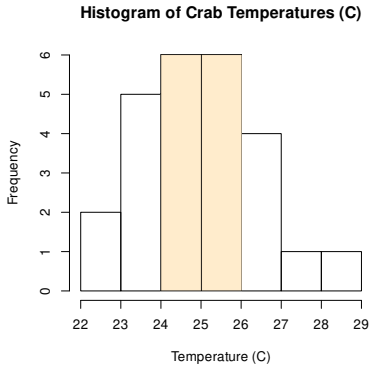


Figure 2.1: For the binned data of Table 1.3, the bins from $24°\text{C} \to 26°\text{C}$ contain the most data points, and hence this interval of numbers is the mode *for this definition of the bins.*

Limitations of the Mode

While the mode may describe an important feature of a data-set, it cannot by itself be a complete indication of what is "typical" of a general data-set, for several reasons.

The first reason becomes apparent if one determines the mode of the data set $\{1, 1, 9, 10, 11, 12, 13, 14\}$:

$$\underbrace{1 \quad 1 \quad 9 \quad 10 \quad 11 \quad 12 \quad 13 \quad 14}$$

Is the mode= 1 really representative of the "typical" datum?

The mode is 1, but the majority of the data are 9 or larger. Saying *"a typical datum is 1"* paints a very misleading picture of the data.

There is no standard function in R to compute the mode of a vector. It is possible to write a sequence of commands that finds the mode, but we won't need to do so.

A second reason the mode is limited is that there may not be unique mode unless you bin the data. For example, in the data-set {5.23, 7.69, 9.14, 10.3} no number shows up more than once, meaning either that there is no mode, or that there is a 4-way tie for the mode. Neither of these possibilities provides much insight into the data. While binned data does have a mode, it isn't unique, because there are many different ways to define the bins.

Of the three common definitions of *average*, the mode is by far the least commonly used. This is why there is no R function for it. However, we will revisit the idea of the mode in a rather different context in Sec. 16.7.

For these reasons, it is important to look for other ways to characterize a "typical" datum.

2.2 The Median

Definition

Rather than looking at which datum values are most frequent, we could also approach the question of "typical" by listing the data in order, and identifying the middle datum. For example, in the data set $x = \{1, 5, 10, 12, 17\}$, the middle datum value is $x_3 = 10$.

Note that because nominal data can't be ordered, there's no meaningful median for such data.

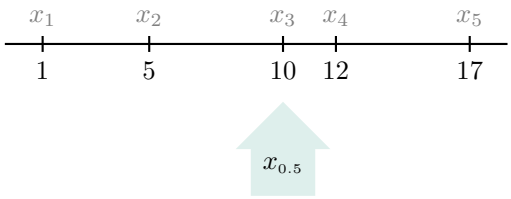


Figure 2.2: The median $x_{0.5}$ is the datam in the middle, i.e. the 3rd datum out of 5. There are 2 data below it, and 2 data above it.

This notion of *typical* or *average* – the datum value with an equal number of data below it and above it – is called the `median` of the data. We will denote the median with a subscript of 0.5, e.g.

$$x_{0.5} = 10$$

A useful feature of the median is that it is not affected by datum values at the low or high-end of the data. The mode is forced to choose a low or high value if it has the highest frequency, as we saw earlier. Furthermore, the mean (discussed next) can be radically changed by a *single* low or high value in the data, while the median will be unchanged. Put succinctly, the median isn't affected much by outlier data.

The median is computed in R using the `median()` function. For our current example, we issue the commands

```
x <- c(1, 5, 10, 12, 17)
median(x)
```
(R.2.1)

and the value of 10 is returned.

Median For An Even Number of Data

In situations where there are an even number of data, the median is defined to be the half-way point (i.e. the *mean*, see Sec. 2.3) of the middle two data. For

example, if we add the datum 8 to the the previous data set, there are now an even number of data. The middle two data are 8 and 10, and the halfway point between them is 9. Thus, the median for this data is 9.

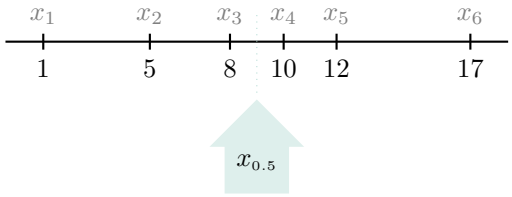


Figure 2.3: For an even number of data, the median is halfway between the middle two data.

Note that an odd number of *ordinal qualitative* data (i.e. qualtitative data you can put in an order) has a well-defined median, but an even number of such data does not usually have a well-defined median, since usually there's no notion of "halfway" between two qualitative data.

Unlike the case for an odd number of data, in an even number of data the median doesn't actually correspond to any individual datum – instead, it is a value halfway between the two innermost datum values.

Limitations

The median's insensitivity to low/high data values can be useful, but can also be problematic. Imagine that five people have the following amount of money in dollars: $M = \{0, 0, 5, 1000, 1000\}$. Of the five people, two are broke, and two have a thousand dollars. The median $M_{0.5} = \$5$ is designed to ignore the poor people and wealthy people (i.e. most of the people), focusing instead on the small minority of people "in the middle". It is thus desirable to also have available a notion of *average* that is influenced by low and high values.

2.3 The Mean

Imbalance Around The Median

Continuing the discussion of the limitations of the median, consider the data set $x = \{5, 7, 12\}$. The median of this data set is $x_{0.5} = 7$. But note that the data are not "balanced" around this median: $x_3 = 12$ is 5 *greater* than 7, but $x_1 = 5$ is only 2 *less* than 7. The data is "top-heavy" around the median of 7. To visualize this, note that in Fig. 2.4 the left-pointing red arrow doesn't match the right-pointing green arrow.

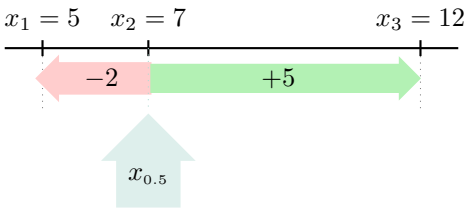


Figure 2.4: Data is "top-heavy" around the median $x_{0.5} = 7$.

Put another way, the differences between the data x_i and the median $x_{0.5}$ don't add up to zero:

$$\begin{aligned}\sum_{i=1}^{3}(x_i - x_{0.5}) &= \underbrace{(x_1 - x_{0.5})}_{5-7=-2} + \underbrace{(x_2 - x_{0.5})}_{7-7=0} + \underbrace{(x_3 - x_{0.5})}_{12-7=+5} \\ &= (-2) + (0) + (5) \\ &= +3 \end{aligned} \tag{2.1}$$

Definition of the Mean

The `mean` of the data x is defined to be the number $\bar{x}$ around which these differences DO add up to zero. The difference betwen a datum x_i and the mean $\bar{x}$ is called the `residual` of the datum, and is denoted R_i:

$$R_i \equiv x_i - \bar{x} \tag{2.2}$$

Thus, our definition for the mean $\bar{x}$ can be expressed as

Because the concept of the mean relies upon the notion of a residual, which is inherently *quantitative*, the mean is not well-defined for *qualitative* data.

$$\sum_{i=1}^{n} R_i = \sum_{i=1}^{n} (x_i - \bar{x}) = 0 \tag{2.3}$$

and this formula allows us to compute the value of the mean. Let's write this out in more detail for our data $x = \{5, 7, 12\}$:

$$\begin{aligned} 0 &= \sum_{i=1}^{3} R_i \\ &= R_1 + R_2 + R_3 \\ &= (x_1 - \bar{x}) + (x_2 - \bar{x}) + (x_3 - \bar{x}) \\ &= (5 - \bar{x}) + (7 - \bar{x}) + (12 - \bar{x}) \\ &= 5 - \bar{x} + 7 - \bar{x} + 12 - \bar{x} \\ &= 5 + 7 + 12 - \bar{x} - \bar{x} - \bar{x} \end{aligned} \tag{2.4}$$

and thus

$$0 = 5 + 7 + 12 - 3\bar{x} \tag{2.5}$$

We can add $3\bar{x}$ to both sides to obtain

$$3\bar{x} = 5 + 7 + 12 \tag{2.6}$$

and dividing both sides by 3 we obtain

$$\bar{x} = \frac{5 + 7 + 12}{3} = 8 \tag{2.7}$$

In Fig. 2.5 we can see that $\bar{x} = 8$ satisfies Eq.(2.4), because the sum of the left-pointing red arrows balances out the right-pointing green arrow:

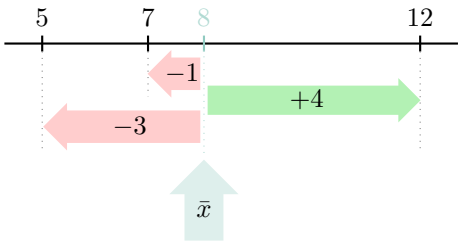


Figure 2.5: Residuals $R_1 = -3$, $R_2 = -1$, and $R_3 = +4$ add to zero.

Computing The Mean

Looking at Eq.(2.7) we see that the mean is computed by adding up the data, and dividing by the number of data points. A general formula for computing the mean is thus:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{2.8}$$

which is sometimes abbreviated to $\bar{x} = \frac{1}{n}\Sigma x$. The mean of a vector in R is computed via `mean()`. Thus, the commands

```
x <- c(5, 7, 12)
mean(x)
```
(R.2.2)

will return the value of 8.

While Eq.(2.8) is useful for computing $\bar{x}$, Eq.(2.3) more directly expresses the *meaning* of $\bar{x}$.

Spreading Things Out Evenly

In addition to the mean representing the point about which the data is neither "top-heavy" or "bottom-heavy", the mean can equivalently be thought of as spreading a total evenly among some number of things. For example, if the data in Fig. 2.5 represented the amount of money 3 people had – \$5, \$7, and \$12 – then the mean $\bar{x} = \$8$ is the amount each person would have if the total \$24 was spread out evenly among the 3 people.

Alternatively, if you take the number of people $n = 3$ and multiply by the mean $\bar{x} = \$8$, you get the total amount of money present. Generally,

$$\text{number} \times \text{mean} \;=\; \text{total} \tag{2.9}$$

The definition of $\bar{x}$ in the form of Eq. (2.3) most directly expresses the idea that $\bar{x}$ is the point about which the residuals are balanced. On the other hand, the definition of $\bar{x}$ in the form of Eq. (2.8) most directly expresses the idea that if you spread out the total into n equal piles then $\bar{x}$ is the amount in each pile.

or equivalently

$$n\bar{x} = \sum_{i=1}^{n} x_i \tag{2.10}$$

which is just Eq.(2.8) if you multiply both sides by n.

Limitations

The two defining features of the mean (balanced residuals and equal distribution of the total) end up being very useful mathematical properties for a wide variety of calculations. These properties are possible because the formula for the mean incorporates *all* of the datum values in the dataset. However, this can be problematic, as *the major weakness of the mean is sensitivity to outliers*. Imagine a group of 99 people with a mean yearly income of \$20,000. If we add a single billionaire to the group, suddenly the mean income is over ten million dollars! This clearly isn't describing what's going on with a *typical* person in the group.

2.4 So Which One Is The "Typical" Datum?

Will the real average please stand up?

We have introduced three notions of "typical": the mode, median, and mean. Each of these is called an *average*. When the news reports that *"The average income in the United States is . . ."*, they could be referring to a mode, median, or mean. In this course we will usually be more precise, and instead of using the word "average" we will use the words mode, median, or mean. When we do say "average" we are referring to the mean.

Most of the time the mode, median, and mean of a dataset are different. For example, for the dataset at the start of the chapter, $x = \{5, 10, 10, 15, 20, 25\}$, you should verify that the mode is 10, the median is $x_{0.5} = 12.5$, and the mean is $\bar{x} = 14.2$.

All three definitions of "typical" or "average" have both strengths and limitations that depend upon the situation, and no single type of average provides a complete characterization of "the typical datum".

2.5 Exercises

1. Make a table showing the general advantages and limitations of the 3 definitions of average.

2. A group of 20 water samples has mean e-Coli counts of 0.2 (in units of colonies per 100 mL). A 21st sample with e-Coli count of 370 is added to the group. What is mean of the new group of 21 samples? Explain whether or not you feel this mean describes a *typical* sample, and indicate why.

3. Consider the pigeon data from Section 1.5, Exercise 2.

   i) Use R to compute the median and mean of the data. Use a histogram to infer the mode.

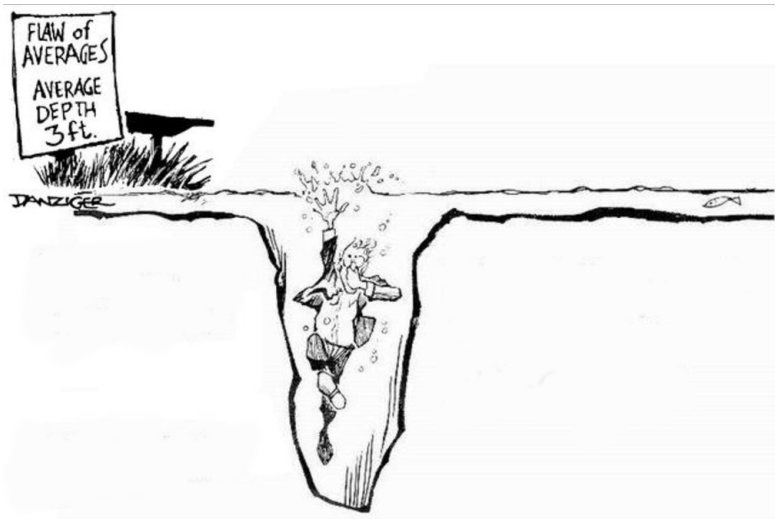
   ii) Give a formula that would allow one to convert a datum value from mm to inches.

   iii) Use your formula from ii) to predict and explain precisely (mathematically) what will happen to the median, mean, and histogram of the data if it were expressed in inches. You may find it helpful to recall some of the ideas of Exercise #4 from Sec. 1.5.

   iv) Use R to convert the data from mm to inches. Use R to compute the new mean and median, and make a new histogram. Do these line up with your predictions from iii)?

4. Consider the following data, representing the wealth (in dollars) of a group of individuals: 1, 1, 3, 5, 15, 29, 37, 568, 1530000. Compute the mean and median using R, and identify the mode. Indicate the strengths and weaknesses of each type of average in characterizing the "typical wealth" of the group. For each type of average, if this is reported as a "typical wealth" of the group, what type of picture is painted for the audience?

5. For the person shown in the comic below, what went wrong with their understanding of the idea of "average"?



This comic was drawn by Jeff Danziger, and appeared in Sam Savage's piece "The Flaw of Averages" from the Oct. 8, 2000 Sunday edition of the San Jose Mercury News [5].

CHAPTER 3

Deviations From The Mean

3.1 An Actual Conversation. Really.

This was not an actual conversation. The author is a liar.

Teacher: Take a look at the two datasets in Fig. 3.1.

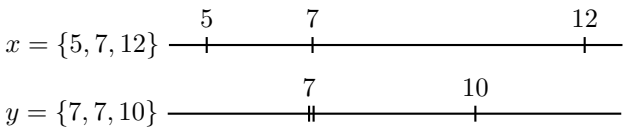


Figure 3.1: Datasets x and y shown on number lines.

Student: I think dataset x looks like the one from Sec. 2.3.

Teacher: That's right. There is also a new dataset y. What are the means $\bar{x}$ and $\bar{y}$ of the two datasets?

Student: Hmm. Give me a moment. *(thumbs through notes)* Using Eq. (2.8), the mean of x is

$$\bar{x} = \frac{x_1 + x_2 + x_3}{3} = \frac{5 + 7 + 12}{3} = 8,$$

and the mean of y is

$$\bar{y} = \frac{y_1 + y_2 + y_3}{3} = \frac{7 + 7 + 10}{3} = 8.$$

Huh. I guess the datasets have the same mean.

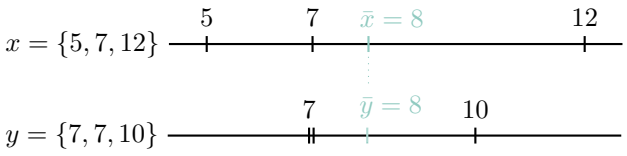


Figure 3.2: Datasets x and y have the same mean.

Teacher: Yes. But the mean doesn't tell the whole story. There is something fundamentally different about x and y. What do you notice about how

"spread out" the data are from the mean in each dataset? In other words, do you notice anything about how the residuals compare in each dataset?

Student: *(thinks for a while)* I'm not sure. Let me draw a picture.

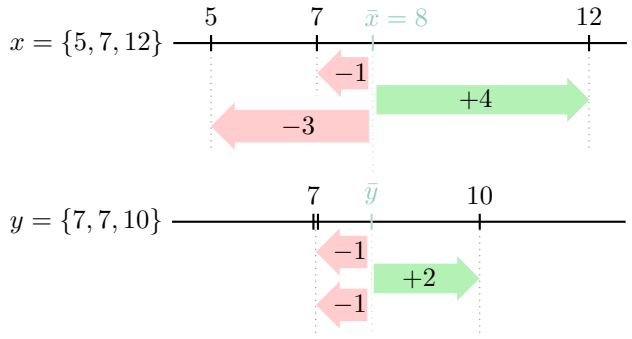


Figure 3.3: Dataset x and y have the same mean, but the y data are huddled closer to the mean, while the x data are more spread out from the mean.

The residuals add to zero in both datasets. But I guess the residuals in x look bigger than those in y?

Teacher: That's right. The datum values in x are spread out farther from the mean than they are in y. How can we *quantify* this larger spread?

Student: Well, we could look at the *average* residual in each dataset. Maybe we should find the mean of the residuals R_i in each case?

Teacher: That sounds reasonable. Go ahead and compute the mean residual $\overline{R}$ for each dataset.

Student: Ok. To compute a mean, you add up the things and divide by the number of them. So I guess you can do that with the residuals R_i. So for x the mean residual is

$$\overline{R} = \frac{\sum R_i}{n} = \frac{(-1) + (-3) + (4)}{3} = 0.$$

Zero? *(thinks a bit)* Oh right. By the definition in Eq. (2.3), the residuals always add up to zero, and hence the mean residual is zero. The sum of the negative residuals always balances the sum of the positive residuals. That's basically the definition of the mean.

Teacher: Good thinking. Perhaps we should just pay attention to the *lengths* of the bars in Fig. 3.3, and not pay attention to the plus or minus sign.

Student: Get rid of the minus? That seems familiar. *(thumbs through algebra book)* Oh yeah. That's absolute value. The absolute value makes a number positive, so we could take the absolute value $|R|$ of each residual first, and then find the mean $\overline{|R|}$. For dataset x, we have

$$\text{dataset } x\text{:} \quad \overline{|R|} = \frac{|-1| + |-3| + |4|}{3} = \frac{1 + 3 + 4}{3} \approx 2.67$$

and for dataset y we get

$$\text{dataset } y\text{:} \quad \overline{|R|} = \frac{|-1| + |-1| + |2|}{3} = \frac{1+1+2}{3} \approx 1.33$$

Yeah – that seems to work. We get a bigger number for the spread of dataset x, which is just what we see in Fig. 3.3. So $\overline{|R|}$ seems like a good way to quantify the spread.

Teacher: Nice work. $\overline{|R|}$ is a valid way of expressing *variation* of the data about the mean.

Here's a bummer though – and I hate to do this to you – but absolute value doesn't work well in the advanced math (like calculus) statisticians use. So we need to find another way to quantify the spread.

Remember from algebra that squaring a number (multiplying it by itself) is another way to convert any number into a positive number (though different in size), because a negative times a negative is positive. And unlike absolute value, squaring numbers works well with advanced math.

Student: I remember that from algebra – when you square a number, the result is always positive. So you're saying we should take an average of the square of the residuals? Like $\overline{R^2}$?

Teacher: It's worth a try!

Student: Well, for dataset x, the residuals are $R_1 = -1$, $R_2 = -3$, and $R_3 = 4$. If I square them I get $R_1^2 = (-1)^2 = (-1) \times (-1) = 1$, and likewise $R_2^2 = (-3) \times (-3) = 9$, and $R_3^2 = 4^2 = 16$. So the average of the residuals squared is

$$\text{dataset } x\text{:} \quad \overline{R^2} = \frac{R_1^2 + R_2^2 + R_3^2}{3} = \frac{1+9+16}{3} \approx 8.67$$

and doing the same thing for y gives

$$\text{dataset } y\text{:} \quad \overline{R^2} = \frac{R_1^2 + R_2^2 + R_3^2}{3} = \frac{1+1+4}{3} = 2.$$

That seems good - when computing the spread this way, we got a bigger number for the $\overline{R^2}$ of dataset x, and that agrees with the bigger spread we see in Fig. 3.2. Please tell me this is what we're looking for – this seems to be going on forever!

Teacher: No worries - you did good! No more questions for you. Until next time, that is . . .

3.2 The Variance s^2

The previous discussion indicates that we can quantify the spread of data about the mean by squaring each residual, and then taking an average:

$$\text{spread around } \bar{x} = \overline{R^2} = \frac{1}{n}\sum_{i=1}^{n} R_i^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \bar{x})^2$$

It turns out that we need to make a slight modification to this formula: instead of dividing by n, we should divide by $n-1$, for reasons discussed in Comment Box 3.1. The resulting quantity, denoted as s^2, is known as the `variance` of the data:

$$\text{variance } = s^2 \equiv \frac{1}{n-1}\sum_{i=1}^{n} R_i^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1} \tag{3.1}$$

The symbol s^2 is due to the variance being the square of a more commonly used quantity, the standard deviation s, discussed in the next section.

The variance is computed for a vector in R with the function `var()`.

Comment 3.1 – Why $n-1$? (Optional Reading)

The reason is subtle and not obvious at all, but here is the rough explanation. Consider the dataset $x = \{5, 7, 12\}$ discussed above. When we sit down to compute $\bar{x}$, note that each of the three datum values x_i does not depend upon the others – they are three independent measurements that have no relationship with each other. So we divide by 3.

This isn't true when computing the mean of R^2. For example, if somebody told you *"You can't see the data, but there are 3 datum values, and the first two residuals are $R_1 = -1$ and $R_2 = -3$. What is R_3?"* you can answer the question. R_3 must equal 4, because the residuals must add to zero. In other words, because $R_1 + R_2 + R_3 = 0$, one of the residuals is NOT independent of the other two. So when computing means involving the residuals themselves, we should divide by 2, or in general, $n-1$.

Ultimately, the justification for using the $n-1$ factor, known as *Bessel's Correction*, is that it allows for what statisticians call an *unbiased estimator*. Roughly, this means if we use Eq. (3.1) on a huge number of different sample datasets, the results will average out to the "correct value" for a population, though we can't be more precise until we discuss probability distributions in Part II of the book. At any rate, it's worth noting that there are arguments against always using *Bessel's Correction*. For example, see [6] Jayne's *Probability Theory: The Logic of Science*, Section 17.2 for a discussion of how an unbiased estimator isn't necessarily an optimal estimator, in the sense of reducing the overall expected error of an estimate for a single sample.

Example 3.1: Computing Variance

Problem: The masses of 4 samples are recorded as $x = \{2.1, 4.3, 6.9, 10.1\}$ kilograms. Compute the variance of x by hand, and verify using R.

Solution: First we compute the mean of the $n = 4$ datum values. By Eq. (2.8)

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{2.1 + 4.3 + 6.9 + 10.1}{4} = 5.85 \text{ kg}$$

Once we know the mean we can compute the residuals using Eq. (2.2):

$$\begin{aligned} R_1 &= (x_1 - \bar{x}) = (2.1 - 5.85) = -3.75 \text{ kg} \\ R_2 &= (x_2 - \bar{x}) = (4.3 - 5.85) = -1.55 \\ R_3 &= (x_3 - \bar{x}) = (6.9 - 5.85) = 1.05 \\ R_4 &= (x_4 - \bar{x}) = (10.1 - 5.85) = 4.25 \end{aligned}$$

and finally we can square each residual and use Eq. (3.1) to compute the variance:

$$s^2 = \frac{\sum_{i=1}^{n} R_i^2}{n-1} = \frac{(-3.75)^2 + (-1.55)^2 + (1.05)^2 + (4.25)^2}{4-1} \approx \boxed{12 \text{ kg}^2}$$

While we computed $s^2 = 11.87667$, for real data it would be inappropriate to report our result with that many decimal places, since the original data were only given with two significant figures.

Checking the result in R, we enter the command `x <- c(2.1,4.3,6.9,10.1)` and then `var(x)` gives `11.87667`, which rounds to our answer.

3.3 The Standard Deviation

One drawback of the variance s^2 is that it doesn't have the same units of measurement as the original data. In Example 3.1, the original data are in kg, but the variance has units of kg^2. It's not easy to visualize what a square-kilogram means, and hence it is tough to get a feeling for the actual amount of spread in the data.

The solution is to take a square root of the variance, so that we're back to the original units of measurement. The square root of the variance is called the `standard deviation`, is denoted by the letter s,

$$s = \sqrt{\text{variance}} = \sqrt{s^2} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} = \sqrt{\frac{\sum_{i=1}^{n}(x-\bar{x})^2}{n-1}} \tag{3.2}$$

and may be computed for a vector in R with the function `sd()`. The standard deviation provides a measure of the spread (i.e. variation) that is easier to visualize.

Example 3.2: Computing Standard Deviation

Problem: Compute and interpret the standard deviation s for the data in Example 3.1.

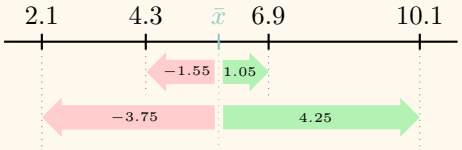
Solution: We computed that the variance was $s^2 = 11.87667\ \text{kg}^2$. Taking a square root we get

$$s = \sqrt{s^2} = \sqrt{11.87667} \approx \boxed{3.45\ \text{kg}}.$$

Continuing with the R commands of Example 3.1, we can check our work by issuing the command `sd(x)`.

The standard deviation has units of kg, just like the original data. The interpretation of $s = 3.45$ kg is that most of the data aren't much farther than this from the mean $\bar{x} = 5.85$, but also most of the data aren't a whole lot closer (like within 0.1 or something).

Put another way, we wouldn't expect many of the residuals to be a whole lot bigger or a whole lot smaller than 3.45. This is consistent with the residuals computed in Example 3.1, which we visualize below:



Thus, most of the datum values are clustered around $\bar{x} = 5.85$ kg within a distance that is roughly $s = 3.45$ kg. We will learn to be more precise in Part II of the book.

It's not as accurate to take a square root of our rounded answer $s^2 = 12$, because taking the square root of a rounded number could introduce additional rounding errors. In this case, $\sqrt{12}$ is 0.5% larger than the true answer. Not a big difference, but in an analysis involving lots of steps, such rounding errors can start to add up.

In general, it is best to keep more digits *inside* a calculation, but then at the end report the overall result with an appropriately reduced number of significant digits.

3.4 Exercises

1. For the pigeon data in the Ch. 1 exercises, please use R where appropriate to do the following, and include your source code and plots:

   i) Compute the variance s^2 and standard deviation s.

   ii) In your own words, describe the interpretation of s for this dataset.

   iii) Plot the histogram, and draw red vertical lines at $\bar{x} + s$ and $\bar{x} - s$ via the `abline()` function and `col` option. You may want to find some examples of how to do this online.

   iv) Make a new data set by adding 4 mm to all of the original data. What is the new mean and standard deviation? Plot the new histogram. How is it similar to the original histogram, and how is it different?

   v) Make yet another new data set by multiplying all of the original data by 2. Again, what is the new mean, standard deviation, and histogram?

2. Consider a data-set with data x_i with mean $\bar{x}$ and standard deviation s.

   i) A positive number c is added to all of the data points. How does this affect the mean and standard deviation of the data? What will happen to the histogram of the data?

   ii) Now instead imagine that the original data is multiplied by a positive number c. Again, what is the affect on the mean, standard deviation, and histogram?

   You do not need to prove your answers using Σ calculations (though of course you're free to do so if you want to stretch your algebra skills). However, you should explain your answers.

3. As a summer research project, you are studying two different populations of the beetle *Scarabaeus NotRealica*. Population A is located on the tropical island of Statistica, and population B is located on the tropical island of Quantifica. You find as many beetles as you can, and record the following masses (in grams):

   Population A:

   $$M_A = \{15.46, 14.71, 16.59, 14.42, 10.68, 13.53, 14.49, 15.46, 17.59, 12.31\}$$

   Population B:

   $$M_B = \{16.71, 23.13, 33.30, 22.81, 21.49, 18.57, 14.26, 11.33, 18.76, 15.60, \\ 19.92, 14.07, 7.32, 9.64, 22.98, 18.86, 9.46\}$$

   Using the ideas presented up through this chapter, compare the two datasets M_A and M_B: in what ways are they similar? In what ways are they different? What do these differences and similarities imply about the beetles at the two locations? What might be some plausible causes for these types of differences?

CHAPTER 4

Quantiles and Box Plots

In this chapter we discuss how the concept of the median can be extended to the more general idea of a *quantile*. The notion of quantiles will be primarily used in two ways as regards descriptive statistics. One, to offer an alternative way to describe the spread in a dataset, known as the *interquartile range*. Two, to define the *boxplot*, which is a quick method of visualizing a dataset. We will revisit quantiles in a slightly different context when we discuss probability theory in Part II.

4.1 Definition

Extending The Concept of the Median

In Sec. 2.2 we defined the median value $x_{0.5}$ as the number such that half of the datum values were less than or equal to $x_{0.5}$. Because a fraction $1/2 = 0.5$ of the datam values are less than the median, we denote it with the subscript of 0.5. As an example, consider the weights w_i of 11 minnows, as shown in Table 4.1. If the datum values are arranged in order on a number line, as in Fig. 4.1, we can see that the datum $w_6 = 14.6$ g is the median $w_{0.5}$, since a fraction 0.5 of the weights are smaller than this number. For reasons that will become clear, let us relabel the median as $q_{0.50}$.

minnow #	weight (g)
1	2.1
2	5.2
3	7.5
4	9.3
5	11.0
6	14.6
7	18.4
8	20.7
9	30.9
10	34.2
11	97.1

Table 4.1: Minnow weights

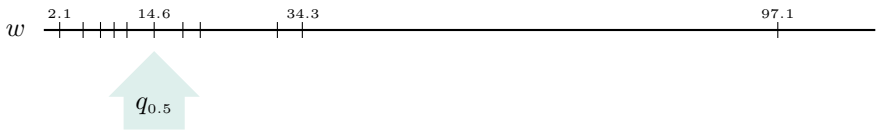


Figure 4.1: Values from Table 4.1 shown on a number line, with median indicated.

The concept of the median can easily be extended to other fractions between 0 and 1. Just as the median $q_{0.50}$ is defined by having a fraction 0.5 of the datum values below it, the quantity $q_{0.75}$ is the value such that a fraction 0.75 of the datum values are below it, as shown in Fig. 4.2.

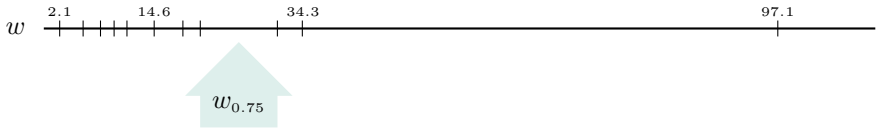


Figure 4.2: The 0.75 quantile $q_{0.75}$ is $w = 25.8$. For this dataset, $q_{0.75}$ falls 50% of the way between w_8 and w_9, for reasons we will explain below.

The value $q_{0.75}$ is an example of a `quantile`. The quantile q_p of dataset w is the value such that a fraction p of the datum values in w are below it.

The use of the letter p to denote the fraction between 0 and 1 is related to the concept of *probability*, which is discussed in Part. II.

How To Compute A Quantile

Sometimes the sorted data values w_1, w_2, etc. are referred to as *order statistics*.

Let's examine how the value of $q_{0.75}$ was computed in Fig. 4.2. First make sure the data is sorted in ascending order. Now note that there are $n = 11$ datum values in the dataset. We know that $w_1 = 2.1$ is the 0 quantile, i.e. $q_{0.00}$. There are $n - 1$ datum values past this first one. If we were trying to find the median, i.e. the halfway point, the datum we want is $0.5 \times (n - 1)$ past the first datum, i.e. $0.5 \times 10 = 5$ past the first datum, meaning datum # 6. If we're trying to find $q_{0.75}$, we want to look at the datum that is $0.75 \times (n - 1)$ past the first datum, i.e. $0.75 \times 10 = 7.5$ past the first datum, i.e. datum # 8.5. Let's call this value $w_{8.5}$.

We thus need to find the number that's halfway between the 8^{th} and 9^{th} datum values. To compute this, note that $w_9 - w_8 = 30.9 - 20.7 = 10.2$. Thus, if we start at $w_8 = 20.7$ and then add on another 10.2 we get all the way to w_9. However, if we only add 0.5×10.2 then we end up half-way between them. Thus, $w_8 + 0.5 \times (w_9 - w_8)$ will get us the quantile we want:

$$\begin{aligned} q_{0.75} &= w_{1+0.75(n-1)} = w_{1+0.75\times 10} = w_{8.5} \\ &= w_8 + 0.5 \times (w_9 - w_8) \\ &= 20.7 + 0.5 \times (30.9 - 20.7) \\ &= 25.8 \end{aligned}$$

We can generalize this procedure as follows. To find the p quantile of a dataset x with n values, we first compute the quantities

In math the notation $\lfloor x \rfloor$, called the *floor function*, just means round down. Thus, $\lfloor 5.26 \rfloor = 5$, etc.

$$\alpha \equiv \lfloor 1 + p(n-1) \rfloor \tag{4.1}$$

$$\beta \equiv \text{frac}\,(1 + p(n-1)) \equiv 1 + p(n-1) - \lfloor 1 + p(n-1) \rfloor \tag{4.2}$$

where α denotes the integer part of $1 + p(n-1)$, i.e. the value rounded down, and β is the fractional part of $1 + p(n-1)$, i.e. the part after the decimal.

Recalling the example above, we had $1 + p(n-1) = 1 + 0.75(11-1) = 8.5$, so in this case $\alpha = 8$ and $\beta = 0.5$.

If $\beta = 0$ then we are done, as the datum value x_α is the quantile in question. If $\beta \neq 0$ we compute the p-quantile as being a fraction β between the α^{th} and $(\alpha+1)^{\text{th}}$ datum values:

$$q_p = x_\alpha + \beta \times (x_{\alpha+1} - x_\alpha) \tag{4.3}$$

where α and β are defined in Eqs. (4.1),(4.2).

Percentiles

We will typically describe a quantile with a number between 0 and 1. Hence, the median $q_{0.5}$ is the 0.5 quantile of a dataset. However, we can also express the decimal 0.5 as 50%. When the quantile is expressed using percents, it is called a `percentile`. Therefore the median $q_{0.5}$ is the 50^{th} percentile $q_{50\%}$. Percentiles are sometimes denoted with a capital P, e.g. the median is P_{50}, $q_{0.12}$=P_{12} etc.

Reminder that to convert a decimal to a percentage you multiply by 100, or equivalently move the decimal place left two spaces. To go from a percent to a decimal, divide by 100 or move the decimal place left two spaces.

Quartiles

The quantiles at 0.25, 0.5, and 0.75 are also known as the 1^{st}, 2^{nd}, and 3^{rd} `quartiles`, since they divide the data into quarters (or 4 parts), and are often denoted with a capital Q. The quartiles of a dataset are defined as

$$1^{\text{st}} \text{ quartile } = Q_1 = q_{0.25} = q_{25\%} = P_{25} = \text{ lower quartile} \tag{4.4}$$

$$2^{\text{nd}} \text{ quartile } = Q_2 = q_{0.50} = q_{50\%} = P_{50} = \text{ median} \tag{4.5}$$

$$3^{\text{rd}} \text{ quartile } = Q_3 = q_{0.75} = q_{75\%} = P_{75} = \text{ upper quartile} \tag{4.6}$$

Example 4.1: Determining Quartiles

Problem: Lead concentrations (in parts per billion) are measured at different locations in the Missouri river, as shown in Table 4.2. Compute the quartiles of the data.

location #	Pb (ppb)
1	8.3
2	10.8
3	12.5
4	14.1
5	42.1
6	6.9
7	25.3
8	11.4
9	15.3
10	21.1
11	16.0
12	22.2
13	25.8
14	8.3

Table 4.2: Lead concentrations in Missouri River locations.

Solution: For this example let's compute the quartiles without worrying about the corresponding locations in the river. We sort the lead data and for convenience label the sorted lead values with the letter L.

We compute the quartiles using Eqs. (4.1),(4.2), and (4.3). Noting that there are 14 datum values, i.e. $n = 14$, for the lower quartile $q_{0.25}$ we compute $1+p(n-1) = 1+0.25(14-1) = 4.25$, meaning $\alpha = 4$ and $\beta = 0.25$. We want the value that is a fraction 0.25 between the *sorted* datum values L_4 and L_5:

$$\begin{aligned} q_{0.25} &= L_\alpha + \beta \times (L_{\alpha+1} - L_\alpha) \\ &= L_4 + 0.25 \times (L_5 - L_4) \\ &= 10.8 + 0.25 \times (11.4 - 10.8) = 10.95 \end{aligned}$$

and thus $\boxed{Q_1 = q_{0.25} = 11.0 \text{ ppb}}$.

sorted #	L (ppb)
1	6.9
2	8.3
3	8.3
4	10.8
5	11.4
6	12.5
7	14.1
8	15.3
9	16.0
10	21.1
11	22.2
12	25.3
13	25.8
14	42.1

Table 4.3: Sorted data from Table 4.2.

For the middle quartile, i.e. the median, we have $1+p(n-1) = 1+0.5(14-1) = 7.5$, meaning $\alpha = 7$ and $\beta = 0.5$. We want the value that is a fraction 0.5 between L_7 and L_8:

$$\begin{aligned} q_{0.50} &= L_\alpha + \beta \times (L_{\alpha+1} - L_\alpha) \\ &= L_7 + 0.50 \times (L_8 - L_7) \\ &= 14.1 + 0.50 \times (15.3 - 14.1) = 14.7 \end{aligned}$$

so that $\boxed{Q_2 = q_{0.50} = 14.7 \text{ ppb}}$. Note that this matches the instructions of Sec. 2.2: for even n, the median is the number halfway between the two middle datum values.

Finally, for the upper quartile we have $1+p(n-1) = 1+0.75(14-1) = 10.75$ so that $\alpha = 10$ and $\beta = 0.75$. We want the value that is a fraction 0.75 past L_{10} towards L_{11}:

$$\begin{aligned} q_{0.75} &= L_\alpha + \beta \times (L_{\alpha+1} - L_\alpha) \\ &= L_{10} + 0.75 \times (L_{11} - L_{10}) \\ &= 21.1 + 0.75 \times (22.2 - 21.1) = 21.925 \end{aligned}$$

so that $\boxed{Q_3 = q_{0.75} = 21.9 \text{ ppb}}$.

Note that none of the quartiles correspond to actual datum values in the dataset, which is a typical.

4.2 Computing Quantiles in R

Quantiles can be quickly computed in R via the `quantile()` function. Consider the crab temperature data defined in Sec. 1.3.1. If we wish to see the 25^{th}, 37^{th}, 50^{th}, and 80^{th} percentiles of the vector `x`, we collect the quantiles we want into a vector via `c()` in the command

When using `quantile()` you must express the quantiles as numbers from 0 to 1, and not as percentages.

```
quantile(x, c(0.25, 0.37, 0.5, 0.8) )
```
(R.4.1)

where `x` is defined in (R.1.1). If we would like to examine quantiles from 0 to 1 at 20% intervals, we could instead issue the command

`seq(a,b,c)` makes a vector with numbers that start at `a`, end at `b`, in increments of `c`. Hence `seq(2,4,0.5)` is equivalent to the command `c(2,2.5,3,3.5,4)`.

```
quantile(x, seq(0,1,0.2) )
```
(R.4.2)

with results

```
   0%    20%    40%    60%    80%   100%
22.90  23.90  24.60  25.40  26.12  28.10
```
(R.4.3)

Note that the single R command `summary(x)` returns the quartiles, the mean, and the minimum/maximum values for a vector `x`. These numbers (w/o the mean) are called a `five-number summary` of the data.

4.3 Interquartile Range

As A Measure of Spread

One of the many uses of quantiles is that they give us an alternate way of defining the spread in data. Instead of looking at the standard deviation Eq. (3.2), one could look at the distance between the lower quartile $Q_1{=}x_{25\%}$ and the upper quartile $Q_3{=}x_{75\%}$. This difference is known as the interquartile range or IQR:

$$\text{IQR} = \text{interquartile range} = Q_3 - Q_1 = q_{0.75} - q_{0.25} \tag{4.7}$$

The interquartile range is to be distinguished from the simple `range` of a dataset, which is typically defined as the largest datum value minus the smallest datum value, i.e. $q_{1.00} - q_{0.00}$. The problem with the range is that it is very sensitive to outliers, which are discussed on the next page.

The IQR may be computed for a vector in R via the function `IQR()`.

Example 4.2: Computing IQR in R

Problem: Compute and compare the IQR and the standard deviation s for the crab data defined in (R.1.1) in Sec. 1.3.1.

Solution: For the crab data `x`, the interquartile range is computed with `IQR(x)` to give 1.8°C. Equivalently, we could simply compute the 75^{th} and 25^{th} percentiles and subtract them, via

Reminder: at the console you could just type `Q3-Q1` and the result will appear. In an R script, the `print()` function is necessary in order for the result to appear in the console.

```
Q1 <- quantile(x,c(0.25))
Q3 <- quantile(x,c(0.75))
print(Q3-Q1)
```

to again obtain 1.8°C. By comparison, the standard deviation is $s{=}$ `sd(x)` $=$1.34°C. The IQR and s are generally different.

Remember that s numerically incorporates all datum values, while IQR does not. One advantage of the IQR over s is similar to the advantage that the median has over the mean: it doesn't change in the face of outliers.

Defining Outliers

An *outlier* is a datum value that is seemingly very far away from the rest of the data, and hence *might* be an erroneous datum we should ignore. Even though what counts as "very far away" is somewhat arbitrary, it is still useful to have some sort of agreed-upon criterion for what constitutes an outlier. A common criterion is:

Note that while an outlier may just be an error in data collection or some kind of bizarre rarity, outliers may actually have significance in the analysis. Hence, they shouldn't automatically be thrown out of the data, unless there is a compelling reason.

$$\text{outliers} = \text{values at least } 1.5 \times \text{IQR less than } Q_1 \text{ or greater than } Q_3 \tag{4.8}$$

Example 4.3: Identify Outliers

Problem: Identify any outliers in the lead data from Example 4.1.

Solution: Using the results from Example 4.1, the IQR is

$$Q_3 - Q_1 = 21.9 - 11.0 \approx 11.0$$

and according to Eq. (4.8), outliers are defined as

$$\text{datum} < \underbrace{Q_1 - 1.5 \times \text{IQR}}_{11.0-1.5\times 11.0 = -5.5} \quad \text{OR} \quad \text{datum} > \underbrace{Q_3 + 1.5 \times \text{IQR}}_{21.9+1.5\times 11.0 = 38.4}$$

Any datum values less than -5.5 or greater than 38.4 are considered outliers. These two cutoff numbers are called `fences`.

Since lead concentrations can't be negative, there are no possible outliers beneath the lower fence. However, the datum $\boxed{L_{14} = 42.1 \text{ ppb}}$ is an outlier above the upper fence, by the criterion of Eq. (4.8).

sorted #	L (ppb)
1	6.9
2	8.3
3	8.3
4	10.8
5	11.4
6	12.5
7	14.1
8	15.3
9	16.0
10	21.1
11	22.2
12	25.3
13	25.8
14	42.1

Table 4.4: Sorted lead data from Example 4.1.

We emphasize again that a datum value being classified as an "outlier" doesn't automatically mean it is invalid or should be thrown out. Discarding outlier data from an analysis is a complicated judgment call, and in general there should be a good reason to do so.

4.4 Box Plots

A quick way of visually assessing a dataset in relation to its five-number summary is the `boxplot`, which is created in R with the `boxplot()` function. For the data of Example 4.1, the following code creates a boxplot in R:

(R.4.4)
```
L <- c(6.9,8.3,8.3,10.8,11.4,12.5,14.1,15.3,16.0,21.1,22.2,
       25.3,25.8,42.1)
boxplot(L,ylab="Pb (ppb)",lwd=2)
abline(h=seq(10,40,5),lty="dotted",col="gray",lwd=0.5)
```

The `lwd` parameter sets line thickness. The `abline()` command places some helpful horizontal reference lines.

with results shown in Fig. 4.3. The central "box" in the plot has horizontal lines at the quartiles (the median is thicker than the other lines). The smaller horizontal lines just outside the lower and upper quartiles are the minimum and maximum non-outlier values (as determined by Eq. (4.8)). The vertical dashed lines, indicating variation outside the IQR, are called `whiskers`. The dot at the top of the graph is the outlier of 42.1 ppb. If we wish to omit outliers from the plot, include the optional parameter `outline=FALSE` as follows:

(R.4.5)
```
boxplot(L,ylab="Pb (ppb)",outline=FALSE)
```

As we will see in Sec. 8.1.1, one way to quickly compare the descriptive statistics of different datasets is to lay their boxplots next to each other.

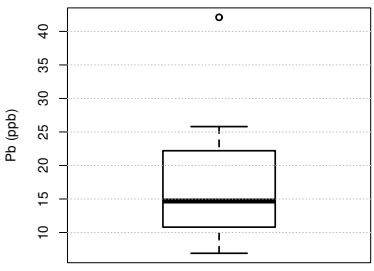


Figure 4.3: A boxplot of the lead data in Example 4.1, with outlier included.

4.5 Exercises

1. Perform the following conversions and show your work:
   i) Convert the decimal numbers 23.4, 0.23, 0.045, and 0.000045 to percents.
   ii) Convert the percentages 0.03%, 45.2%, and 2,867.1% to decimal numbers.

By hand just means don't use statistics software to compute the quantiles. You can still use a scientific calculator to do the arithmetic.

2. For the lead data of Example 4.1, please compute the 40^{th} and 80^{th} percentiles P_{40} and P_{80} of the data *by hand*, showing all steps. Then check your answers with R.

3. Again consider the pigeon data from Section 1.5, Exercise 2 that you revisited in the Ch. 3 exercises. Please use R where appropriate to do the following, and include your source code and plots:
   i) Provide a five-number summary of the data.
   ii) Find the interquartile range and compare it to the standard deviation s.
   iii) What are the cutoff numbers (fences) for a datum to be considered an outlier in this dataset? Are there any outliers in this dataset?
   iv) Use a single command to find all the percentiles between 0% and 100% in intervals of 10%.
   v) Make a labelled box-plot of the data, and explain the meaning of the various lines in the plot.

4. Under what circumstances do you think it would be legitimate to exclude outlier data from an analysis? Feel free to give specific examples to illustrate your points.

5. A dataset has $n = 100$ datum values.
   i) What *fraction* of the datum values lie below $q_{0.20}$? Above $q_{0.65}$?
   ii) What *number* of the datum values lie below $q_{0.20}$? Above $q_{0.65}$?
   iii) What *fraction* of the datum values lie between $q_{0.20}$ and $q_{0.65}$?
   iv) What *number* of the datum values lie between $q_{0.20}$ and $q_{0.65}$?
   v) Redo questions i) thru iv) assuming $n = 250$.
   vi) For a dataset with n values, find a formula that gives the *number* of data points that lie between the quantiles q_{p_1} and q_{p_2}. Your final answer should involve no variables aside from n, p_1, and p_2.

6. An article titled *Comparison of estimated fetal weight percentiles near term for predicting extremes of birthweight percentile* appeared in the American Journal of Obstetrics and Gynecology in Sept. 2020. Explain the meaning of the title.

CHAPTER 5

More About Vectors in R

5.1 Data Selection in Vectors

We often wish only to look at certain parts of a dataset. Let's see how this works in R. We will consider a sample dataset `x` with $n = 10$ data points.

(R.5.1)
```
x <- c(4, 18, 13, 20, 29, 7, -8, 22, 10, -3)
```

5.1.1 Selection Based on Datum Location

We can select certain datum values based on their *location* in the vector. To access only certain elements, we type `x[ ]`, where inside the square brackets we indicate which datum locations we wish to see. Some examples are shown in Table 5.1 below:

To see these elements of `x` ...	type...	and you get
the 3rd	`x[3]`	`13`
the 3rd and 5th	`x[c(3,5)]`	`13 29`
the 3rd repeated	`x[c(3,3)]`	`13 13`
the 3rd through the 6th	`x[c(3,4,5,6)]` or `x[3:6]`	`13 20 29 7`
all of them	`x[]` or `x[1:10]`	whole vector

Table 5.1: Examples of selecting data based on location in vector.

Make special note of the fact that putting *nothing* in the square brackets is equivalent to selecting all elements. This ultimately ends up saving a lot of typing, even though it is more logically consistent for `x[ ]` to return no elements.

Example 5.1: Replacing A Datum Value

Problem: A vector `h` has the value `1.0 9.2 3.4 27 6.6`. The 27 is a mistake though - it should be a 2.7. Fix it!

Solution: Of course we could just redefine the vector `h`. But for large datasets already loaded into R, replacing elements this way can be pretty inefficient. Instead, we note that we can access the 4th element of `h` via `h[4]`. Thus, the code

```
h[4] <- 2.7
```

will replace the 27 by 2.7, so that now `h` is `1.0 9.2 3.4 2.7 6.6`.

Example 5.2: Selecting Data Near Middle

Problem: For the crab temperature data of Sec. 1.3.1, report the 5 datum values in the middle.

Solution: First we find the number of data points using `length(x)` to find that $n = 25$. The middle is the 13^{th} element, and hence we want elements 11 through 15. We first use `sort()` to make sure the data are listed in numerical order, and then select these elements:

```
sort(x)
x[11:15]
```

which returns `24.6 24.8 24.8 25.1 25.4`.

A quick way to compute the middle index is to note that the 1^{st} element has index 1, the last element has index 25, and the mean of these is the index that's in the middle, i.e. $(1 + 25)/2 = 13$. Generally, the middle index is $(1 + n)/2$, which will land on a specific data value if n is odd.

5.1.2 Selection Based on Datum Values

There are many equivalent ways to select vector elements based on their value. In this book we will primarily use two. One way is to use the `subset()` function, which requires you specify the vector and the type of values you want. Another way is to use the square bracket notation from the previous section.

Selection Using The `subset()` Function

One way to select vector elements based on their values is to use the `subset()` function, which takes two arguments:

$$\texttt{subset}\left(\boxed{\begin{array}{c}\text{name of}\\ \text{vector}\end{array}}\ ,\ \boxed{\begin{array}{c}\text{datum values}\\ \text{you want}\end{array}}\right) \tag{5.1}$$

For example, again consider the vector `x` defined in (R.5.1). To select elements of `x` greater than 15, we use the greater-than operator $>$ and use 5.1 as

```
subset(x,x>15)
```
(R.5.2)

with result `18 20 29 22`. Note that the `subset()` function returns a vector we can assign to a new vector name. For example, we could define a new vector (let's call it `xbig`) as

```
xbig  <- subset(x,x>15)
```
(R.5.3)

In addition to using `>` as greater-than, we can use `>=` for greater-than or equal-to, and `==` for equal-to. Statements involving inequality signs or equal signs are known as *conditional statements*, since they place some *condition* on the values we want.

More examples are shown in Table 5.2 below.

To select elements of `x` that are ...	type ...	and get
less than zero	`subset(x,x<0)`	`-8 -3`
greater than 20	`subset(x,x>20)`	`29 22`
greater than or equal to 20	`subset(x,x>=20)`	`20 29 22`
equal to 13	`subset(x,x==13)`	`13`

Table 5.2: Examples of selecting vector elements of `x` based on their values using conditional statements in the `subset()` function.

We can combine conditional statements by putting them in parentheses, and separating them with either the *and* operator `&` or the *or* operator `|`. For example, to select elements of `x` greater than 0 *and* also less than 20, we use the conditional statement `(x>0)&(x<20)` inside of `subset()` as

When multiple conditional statements are combined is this way, the overall result is called a *compound conditional* statement.

(R.5.4)
```
subset(x,(x>0)&(x<20))
```

with result `4 18 13 7 10`. Some more examples are shown in Table 5.3 below:

To select elements of `x` ...	type ...	and get
> 10 **and** ≤ 20	`subset(x,(x>10)&(x<=20))`	`18 13 20`
≥ 10 **and** ≤ 20	`subset(x,(x>=10)&(x<=20))`	`18 13 20 10`
< 0 **or** > 20	`subset(x,(x<0)|(x>20))`	`29 -8 22 -3`

Table 5.3: Examples of selecting vector elements of `x` based on their values using compound conditional statements in the `subset()` function.

Selection Using Bracket [ ] Notation

Another way of selecting elements is to insert the conditional statements inside the square brackets of Sec. 5.1.1. For example, to see the elements of `x` greater than 15, instead of `subset(x,x>15)` one could type

This bracket notation is more compact and saves typing compared to the `subset()` notation, but sometimes it can be harder to read. Feel free to pick the notation you like best and stick with it.

(R.5.5)
```
x[x>15]
```

Table 5.5 below provides a summary of the various selection methods.

To get `x`-values that are ...	type ...	and get
@ 3rd location	`x[3]`	`13`
@ 3rd thru 6th locations	`x[c(3,4,5,6)]` or `x[3:6]`	`13 20 29 7`
@ *all* locations	`x[]` or `x[1:10]`	whole vector
< 0 (less than zero)	`subset(x,x<0)` or `x[x<0]`	`-8 -3`
$>= 20$ (greater than or equal to 20)	`subset(x,x>=20)` or `x[x>=20]`	`20 29 22`
$= 13$ (equal to 13)	`subset(x,x==13)` or `x[x==13]`	`13`
> 10 **and** ≤ 20 (between 10 and 20)	`subset(x,(x>10)&(x<=20))` or `x[(x>10)&(x<=20)]`	`18 13 20`
< 0 **or** > 20 (less than 0 or greater than 20)	`subset(x,(x<0)|(x>20))` or `x[(x<0)|(x>20)]`	`29 -8 22 -3`

Table 5.5: Various ways of selecting elements of `x=c(4,18,13,20,29,7,-8,22,10,-3)` using both bracket notation and the `subset()` function.

90.3	97.9	103.0
82.5	101.8	101.3
86.0	91.9	80.3
80.9	87.2	107.0
89.0	84.5	113.0
78.3	101.7	88.6
98.7	101.0	93.4
55.0	75.3	135.0
87.2	87.2	89.4
90.2	94.5	110.4
91.9	101.6	79.2
82.2	107.7	80.6
68.5	101.1	84.0
100.9	84.6	78.9
89.8	131.0	82.7
113.3	89.7	93.1
96.1	75.5	

Table 5.6: $n = 50$ datum values for mass (grams) of sampled birds. Note that the data here is organized in a grid simply for compactness, and is not a conventional table of the type shown in Table 1.5.

90.3	97.9	103.0
82.5	101.8	101.3
86.0	91.9	80.3
80.9	87.2	107.0
89.0	84.5	113.0
78.3	101.7	88.6
98.7	101.0	93.4
55.0	75.3	135.0
87.2	87.2	89.4
90.2	94.5	110.4
91.9	101.6	79.2
82.2	107.7	80.6
68.5	101.1	84.0
100.9	84.6	78.9
89.8	131.0	82.7
113.3	89.7	93.1
96.1	75.5	

Table 5.7: Outliers indicated for data in Table 5.6.

Example 5.3: Remove Outliers in R

Problem: Use the `subset()` function in R to remove outliers from the data in Table 5.6, according to criterion Eq. (4.8).

Solution: To begin we will start a new R-script, create a vector `x`, and compute both the quartiles Q_1, Q_3 and also IQR. We will use the `unname()` function to remove the label from the quantiles (otherwise it shows a percentile above the number whenever you display it).

```
x <- c(90.3,82.5,86.0,80.9,89.0,78.3,98.7,55.0,87.2,90.2,
       91.9,82.2,68.5,100.9,89.8,113.3,96.1,97.9,101.8,
       91.9,87.2,84.5,101.7,101.0,75.3,87.2,94.5,101.6,
       107.7,101.1,84.6,131.0,89.7,75.5,103.0,101.3,80.3,
       107.0,113.0,88.6,93.4,135.0,89.4,110.4,79.2,80.6,
       84.0,78.9,82.7,93.1)
Q1 <- unname(quantile(x,c(0.25)))
Q3 <- unname(quantile(x,c(0.75)))
xIQR <- Q3 - Q1
```

Now let's compute the lower and upper fences that mark the outliers, according to Eq. (4.8).

```
fence.low <- Q1-1.5*xIQR
fence.high <- Q3+1.5*xIQR
```

which gives 56 and 128 as the fences. Because an outlier is either less than `fence.low` **or** is greater than `fence.high`, we can follow the examples in Table. 5.3 and use an **or** compound conditional in the `subset()` function to see the outlier data:

```
subset(x,(x<fence.low)|(x>fence.high))
```

which returns `55 131 135` as the outliers, as indicated in Table 5.7.

Similarly, non-outlier datum values are both greater than `fence.low` **and** less than `fence.high`, so we can define a trimmed vector with no outliers via an **and** compound conditional:

```
x.trimmed <- subset(x,(x>fence.low)&(x<fence.high))
```

You can compare boxplots to see that `x` has outliers and `x.trimmed` does not.

Note that while you could manually remove outliers in a small dataset, for larger datasets it is much better to remove them via a procedure like the one outlined here. You can also change the outlier criteria. For example, if you only want to exclude data that's more than $5 \times$ IQR away from Q_1 or Q_3 instead of $1.5 \times$ IQR, you could redefine `fence.low` and `fence.high` accordingly.

Finally, even though this procedure illustrates how you *can* remove outliers, this doesn't mean you *should*. In the present dataset, it's true that some birds are much lighter/heavier than the rest, but that by itself doesn't imply the need to exclude this data.

Example 5.4: Fraction of Data In A Range

Problem: For the data in Table 5.6 from Example 5.3, what fraction of the values are within 1 standard deviation s of the mean $\bar{x}$?

Solution: Continuing with the R code from Example 5.3, we can compute the mean and standard deviation via

```
x.mean <- mean(x)
x.sd <- sd(x)
```

We can create a new vector `x.subset` containing only datum values between $\bar{x}-s$ and $\bar{x}+s$. For brevity we will use bracket notation instead of `subset()` notation:

```
x.subset <- x[(x>x.mean-x.sd)&(x<x.mean+x.sd)]
```

and we can find the number of datum values in `x.subset` using the `length()` function:

```
n.x.subset <- length(x.subset)
```

Equivalently, we could skip defining a new vector `x.subset` and combine our previous two operations into a single line as

```
n.x.subset <- length(x[(x>x.mean-x.sd)&(x<x.mean+x.sd)])
```

It usually doesn't matter whether you use `>` or `>=`, etc. Thus, instead of $x > \bar{x} - s$ you could use $x \geq \bar{x} - s$.

The total number of data points is given by

```
n.x <- length(x)
```

and the ratio of these two gives the fraction we are looking for:

```
frac.x <- n.x.subset/n.x
```

which gives $\boxed{39/50 = 0.78}$. Thus, 78% of the data are within 1 standard deviation of the mean.

Note that all of the previous commands can be combined into a single command

```
frac.x <- length(x[(x>x.mean-x.sd)&(x<x.mean+x.sd)])
          /length(x)
```

or, without separately defining `x.mean` and `x.sd`, as

```
frac.x <- length(x[(x>mean(x)-sd(x))&(x<mean(x)+sd(x))])
          /length(x)
```

Once `x` is defined, this last command computes the answer we want in only one line of code! It is perfectly legitimate to combine commands in this way, though it is tougher to read and it's easier to make mistakes. Using more intermediate steps tends to make R scripts easier to read and error-check.

5.1.3 Another Method For Counting Data Subsets

There is a useful alternative to the method of Ex. 5.4. In R you can actually *add* boolean (i.e. true/false) values, where `TRUE` equals 1, and `FALSE` equals zero. For example, the command

```
TRUE + TRUE + FALSE
```
(R.5.6)

returns the value of 2.

We can thus sum of the elements of a boolean vector, and the result will be the number of `TRUE` elements in the vector. As an example, consider the vector

```
x <- c(2,-7,10,4,8)
```
(R.5.7)

Suppose we wished to have R count the number of datum values greater than 5. The command

```
x > 5
```
(R.5.8)

returns the vector

```
FALSE FALSE TRUE FALSE TRUE
```
(R.5.9)

and summing this vector would amount to adding $0+0+1+0+1=2$. In other words, the code

```
sum(x>5)
```
(R.5.10)

Note that this method only *counts* the number of data points meeting some condition. If you also need the actual *datum values*, you have to use the previous methods.

would return 2, which is indeed the number of datum values in `x` that are greater than 5.

Example 5.5: Fraction of Data In A Range, Method 2

Problem: Redo Ex. 5.4 by summing boolean vectors.

Solution: Again we compute the mean and sd as

```
x.mean <- mean(x)
x.sd <- sd(x)
```

The number of values within one sd of the mean is computed via summing boolean values as

```
sum((x>x.mean-x.sd)&(x<x.mean+x.sd))
```

and hence the fraction of values within $\pm s$ of $\bar{x}$ is

```
frac.x <- sum((x>x.mean-x.sd)&(x<x.mean+x.sd))/length(x)
```

or without separately defining $\bar{x}$ and s

```
frac.x <- sum((x>mean(x)-sd(x))&(x<mean(x)+sd(x)))/length(x)
```

5.2 Transforming Data: Changing Units

It is quite common that one wants to change the units of measure of a dataset. For example, say we wanted the data in Table 5.6 from Example 5.3 to be expressed in kilograms instead of grams. Because we have that

$$1 \text{ g} = \frac{1}{1000} \text{ kg} = 0.001 \text{ kg}$$

the equation relating the data in grams and kilograms is

$$\texttt{x.kg} = \texttt{x.g} \times 0.001$$

and thus we can simply multiply all the datum values by 0.001 to get the data in kg:

(R.5.11)
```
x.kg <- x*0.001
```

As another example, consider converting the crab temperature data of Sec. 1.3.1 from °C to °F. Because a celcius temperature C is converted to a Fahrenheit temparture F via the formula

$$F = 1.8C + 32$$

we can convert to °F with the command

(R.5.12)
```
x.F <- 1.8*x+32
```

The conversions shown here are examples of *linear transformations*, being that they simply scale the data by some constant factor and/or add a constant number. More complicated types of data transformation are commonly used as well. In Sec. 28.4 we will examine logarithmic data transformations.

See Sec. 28.1 for a more thorough explanation of what's meant by *linear*.

Example 5.6: Converting Units in Dataset

Problem: The built-in dataset `rivers` gives the lengths of 141 major North American rivers in miles. Convert the lengths to kilometers. What happens to the mean and standard deviation?

Solution: We can look up that 1 mile is approximately 1.609 kilometers. Thus, 2 miles is 2×1.609 kilometers, and x miles is $x \times 1.609$ kilometers. In other words, we multiply the number of miles by 1.609 to get the number of kilometers:

```
rivers.km <- rivers*1.609
```

We could also just rewrite the contents of `rivers` via

```
rivers <- rivers*1.609
```

You should use `mean()` and `sd()` to verify that the mean and standard deviation also increase by a factor of 1.609. It is plausible that this should happen, since all of the data is simply scaled by 1.609, so that the mean and any measures of spread should also scale by this amount. To verify this behavior mathematically, let the mile data values be denoted by x, and note

that the new mean is computed via

$$\begin{aligned}\bar{x}.km &= \frac{1}{n}\sum_{i=1}^{n} x.km_i \\ &= \frac{1}{n}\sum_{i=1}^{n} x_i \times 1.609 \\ &= 1.609 \times \frac{1}{n}\sum_{i=1}^{n} x_i \\ &= 1.609 \times \bar{x}\end{aligned}$$

If you're wondering why the 1.609 came out in front of the summation, remember that if a bunch of things added together are all multiplied by the same number, this number can be factored out. Hence, $ax + ay = a(x + y)$, and $\sum ax_i = a\sum x_i$. This is called the *distributive property*.

which says that the mean in km is 1.609 times as big as the mean in miles. Similarly,

$$\begin{aligned}s.km &= \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x.km_i - \bar{x}.km)^2} \\ &= \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i \times 1.609 - \bar{x} \times 1.609)^2} \\ &= \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}((x_i - \bar{x}) \times 1.609)^2} \\ &= \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2\, 1.609^2} \\ &= \sqrt{\frac{1}{n-1}1.609^2\sum_{i=1}^{n}(x_i - \bar{x})^2} \\ &= \sqrt{1.609^2}\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \\ &= 1.609\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \\ &= 1.609 \times s\end{aligned}$$

We used the property of square roots that $\sqrt{A \times B} = \sqrt{A} \times \sqrt{B}$ to pull the $\sqrt{1.609^2}$ out in front.

In general, if the data is scaled by a factor c, then the mean and standard deviation also scale by this factor.

A related question would be what happens when you not only scale but *shift* the data, as in (R.5.12) above. This is left for you to figure out as an exercise at the end of the chapter.

5.3 Factors and Categorical Data

Creating Factors

Up til now we've dealt with vectors of numbers. However, qualitative data can also be stored in vectors. Consider a datset consisting of eye-color observations:

> brown, brown, brown, brown, brown, amber, blue, brown, amber, hazel, amber, hazel, brown, amber, hazel, green, gray, brown, hazel, blue

What is the best way to store this information in a vector? Well, for starters one could simply create a vector of character strings as follows:

(R.5.13)
```
eye.colors <- c("brown", "brown", "brown", "brown", "brown",
                "amber", "blue", "brown", "amber", "hazel",
                "amber", "hazel", "brown", "amber", "hazel",
                "green", "gray",  "brown", "hazel", "blue" )
```

The problem with `eye.colors` as it currently stands is that R doesn't know there are only 6 possible categories or values of the qualitative data. In other words, R doesn't know the data is *categorical*. As far as R is concerned, "amber" is just a string of characters like any other. Maybe future data could contain the strings "pancake" or "2022 NBA Finals". R's lack of knowledge of the categories makes operations like grouping, sorting, and comparing more difficult. For example, if we try to get some summary information using `summary(eye.color)`, we obtain the following output:

(R.5.14)
```
Length     Class      Mode
    20 character character
```

The *Mode* of a variable or object in R indicates its basic structure, and is something like `numeric`, `character`, `logical`, `list`, `function` etc. The *Class* is a property assigned to an object that determines how various functions operate with it. For example, once we make `eye.color` a factor, its class will be `factor`.

which doesn't tell us much. To let R know our vector is full of categorical data, we use the `factor()` function. In R, a `factor` is just a vector that has categorical data with specified categories. The command

(R.5.15)
```
eye.colors <- factor(eye.colors)
```

converts the original vector into a factor. The different possible values of the category (which R by default just infers from the contents of the vector) are called `levels`. The command `levels(eye.color)` will return the levels (possible datum values) of our factor.

Once a vector is a factor, then many operations in R become easier or more helpful. For example, the command `summary(eye.color)` now results in the output

(R.5.16)
```
amber  blue brown  gray green hazel
    4     2     8     1     1     4
```

which tells us how many data of each level are present.

Level Ordering

Notice that the levels in (R.5.16) are listed in alphabetical order. If you want them to appear in some other order you can use the `levels` parameter of the `factor()` function. For example,

(R.5.17)
```
eye.colors <- factor(eye.colors, levels=c("brown","blue",
                     "hazel","green","gray","amber"))
```

gives a `summary(eye.color)` that appears as

(R.5.18)
```
brown  blue hazel green  gray amber
    8     2     4     1     1     4
```

Up to know we've only been talking about the order that things *appear* in the results. This is all we need for nominal categorical data, but for ordinal categorical data we would like to be able to meaningfully sort and compare data. For example, imagine we had health data for a group of animals:

low, medium, low, low, high, medium, high

If we create a factor with the `ordered` paramater set to `TRUE`, e.g.

(R.5.19)
```
health <- factor(c("low","medium","low","low","high","medium",
                   "high"),
                 levels=c("low","medium","high"),
                 ordered=TRUE)
```

then the categorical data is seen by R as ordinal. We can now make quantitative comparisons. For example, typing `health[1] > health[2]` results in `FALSE`, since the first datum value of "low" is not a greater health than the value of "medium", as determined by our ordering of the levels.

Internal Coding of Factors

It is sometimes useful to know precisely how R stores factor data in a vector. When we print the contents of a factor at the console we see a vector of levels, which are typically character strings, such as "low" or "medium". Internally, however, R regards the data values as integers 1, 2, 3, etc. To see the underlying structure, we can use either the `as.numeric()` or the `unclass()` function. Typing the command

(R.5.20)
```
unclass(health)
```

at the console returns

(R.5.21)
```
int [1:7] 1 2 1 1 3 2 3
 - attr(*, "levels")= chr [1:3] "low" "medium" "high"
```

The reason that `health[1] > health[2]` returns `FALSE` is because 1 is not greater than 2.

which tells us that internally the health values are stored as integers between 1 and 3, where 1 is "low", etc. In Sec. 7.1 we'll see a neat example application of this internal integer coding of factors.

Converting a Factor to Character or Numerical Values

Sometimes it will prove useful to convert a factor back to a vector of character strings, or possibly even numbers. To convert our factor `health` back to character strings, simply use the `as.character()` function:

(R.5.22)
```
health <- as.character(health)
```

Sometimes we wish to treat numerical values as categories, i.e. convert a numerical vector to a factor. If we need to convert back, some caution is needed. For example, suppose we had a reason to define a factor

An example of numerical data being considered categorical might be that you collected 50 soil samples 10 ft from a river bank, another 50 soil samples 20 ft from the bank, and another 50 samples 30 ft from the bank. For some types of analyses this distance from the bank could be considered numerical. In other types of analyses, however, the distance of 10, 20, or 30 ft could be considered a categorical variable used to group the samples.

(R.5.23)
```
x <- factor(c(10.9,3.2,7.6,1.1))
```

If we type `x` at the console, we'll see

(R.5.24)
```
[1] 10.9 3.2  7.6  1.1
Levels: 1.1 3.2 7.6 10.9
```

You may be tempted to think that the internal contents of `x` are the original decimal numbers, but they aren't. Remember that factors are stored internally as integers. We can check this by issuing the command `unclass(x)`, which returns

(R.5.25)
```
[1] 4 2 3 1
attr(,"levels")
[1] "1.1"  "3.2"  "7.6"  "10.9"
```

Accordingly, if we use the `as.numeric()` function by typing `as.numeric(x)` at the console, we'll get the result

(R.5.26)
```
[1] 4 2 3 1
```

which of course is the internal integer coding of the factor levels, and not the original numbers.

So how can we convert the factor back to the original vector of numbers? To proceed, first note that `levels(x)` returns a vector of character-string level values. The commands

(R.5.27)
```
x.chars <- levels(x)
print(x.chars)
```

returns

(R.5.28)
```
[1] "1.1"  "3.2"  "7.6"  "10.9"
```

The elements of this vector can be indexed like any other vector. For example, the command `x.chars[c(2,2,4)]` returns the vector

(R.5.29)
```
[1] "3.2"  "3.2"  "10.9"
```

But remember that internally, `x` is just a vector of integers 1 through 4 indicating the levels, which correspond to the location of the datum value in `x.chars`. Hence, the command `x.chars[x]` yields a vector with the original numbers in character form:

(R.5.30)
```
[1] "10.9" "3.2"  "7.6"  "1.1"
```

and these could be converted back to the original numbers using `as.numeric()`. In summary, to convert a factor `x` back to numerical form, one could use the commands

(R.5.31)
```
x <- levels(x)[x]
x <- as.numeric(x)
```

5.4 Missing Values and NA

It is common for datasets to have missing values at certain datum locations, especially if the datset contains measurements of numerous different variables. In R, missing values are denoted by the logical constant `NA`, meaning "Not Available".

For example, imagine you were studying a pack of 6 wolves, measuring things like weight, sex, etc. However, you were unable to determine the sex of the 2^{nd} wolf. Rather than throw away *all* the data about this wolf, we could instead record a `NA` for the sex of this wolf. Something like

Do not put quote marks around the `NA`. If you do, then R will think that `"NA"` is a character string representing a new factor level.

```
sex <- factor(c("M",NA,"M","F","M","F"))
```
(R.5.32)

Typing `length(sex)` at the console returns 6, meaning that R treats this 2^{nd} position in the vector as existing, but just not holding a numerical value. You can also check by typing `levels(sex)` that `NA` is not considered to be a level.

To remove the NA entries from a vector, we can use the `na.omit()` function:

```
sex <- na.omit(sex)
```
(R.5.33)

Typing `sex` at the console results in

```
[1] M M F M F
attr(,"na.action")
[1] 2
attr(,"class")
[1] omit
Levels: F M
```
(R.5.34)

where the additional information indicates that `NA` values were omitted when creating the object.

NA Isn't NULL or NaN

Sometimes you'll see things like `NULL` or `NaN`, and these aren't the same as `NA`.

In R, `NULL` denotes an empty (nothing there) object. For example, having `NULL` in the 2^{nd} location of a vector will simply *remove* this location from the vector, and the locations past it will fill in the missing space. To see this, type

```
c(2,NULL,3)
```
(R.5.35)

at the console and the output will be

```
[1] 2 3
```
(R.5.36)

Usually you won't run into `NULL` that much, though sometimes you'll see it when trying to import data of unknown data type, or when getting back results from a function that didn't know how to handle the input.

On the other hand, ill-defined numbers in R are denoted by the constant `NaN`, i.e "Not a Number". For example, the operations `0/0` and `sqrt(-2)` both result in `NaN`.

5.5 Exercises

1. Imagine you have the following dataset `x`, where numbers are in mm.

   83.9, 76.8, 67.8, 64.6, 69.1, 61.8, 70.0, 82.4, 81.6, 68.9, 67.3, 77.2, 83.0, 73.1, 42.1, 61.5, 68.7, 65.1, 105.6, 87.6, 59.0, 54.7, 76.2, 78.1, 78.1, 78.5, 84.2, 73.3, 101.0, 60.1, 69.9, 49.7, 85.5, 64.5, 83.4, 81.1, 90.4, 64.0, 92.6, 52.4, 73.0, 80.6, 79.9, 71.3, 82.6, 72.3, 78.8, 96.1, 49.9, 78.1, 78.3, 72.6, 63.0, 92.5, 73.7, 85.7, 52.2, 69.2, 77.6, 76.7, 66.5, 94.6, 76.6, 79.8, 80.2, 77.0, 77.9, 58.9, 92.8, 76.8, 75.7, 72.5, 77.3, 66.8, 75.2, 85.2, 107.9, 73.0, 72.2, 82.2, 68.0, 79.4, 84.1, 77.5, 58.8, 84.4, 66.1, 76.5, 91.8, 58.4, 78.8, 91.1, 93.8, 53.6, 66.6, 68.4, 66.2, 75.5, 84.3, 70.5

   This dataset is posted on the course website for easier import into R.

   Please submit a single R-script that does the following. (You don't have to give results – just have the R-script print the answers to the console, and the instructor will run and check your R script.)

   i) Computes a vector `x.a` containing all elements greater than 90.
   ii) Computes a vector `x.b` containing all elements between 75 and 85.
   iii) Computes a vector `x.c` containing all elements within 10 mm of the mean, and computes what fraction of datum values fall within this range.
   iv) Computes the lower and upper quartiles and the IQR.
   v) Computes the lower and upper outlier fences.
   vi) Computes a vector `x.outliers` containing all outliers.
   vii) Computes a vector `x.trimmed` containing all non-outlier data.
   viii) Creates a labeled boxplot and histogram for both `x` and `x.trimmed`.
   ix) Converts the data from mm to inches.

2. Consider the data transformation shown in (R.5.12). If this conversion is done to a dataset x_{old} to create a transformed dataset x_{new}, what happens to the mean and standard deviation? Specifically, how is $\bar{x}_{\text{old}}$ related mathematically to $\bar{x}_{\text{new}}$, and similarly for s.

3. The ages of 16 sample birds were recorded as Infant (I), Adolescent (A), Young Adult (Y), Middle-Aged Adult (M), Old Adult (O):

   I, A, A, M, M, A, O, I, Y, M, Y, O, M, Y, Y, A

   i) In R create an ordered factor named `age` that holds the datum values.
   ii) What are the levels of `age`?
   iii) Use `unclass()` to indicate the underlying encoding of the level values.
   iv) Use `summary()` to determine the number of birds of each age level.
   v) What is the result of the command `age[5] > age[7]`? Explain how R arrived at this result.
   vi) Use `sort()` to sort the factor in ascending order (from youngest to oldest).
   vii) Convert `age` to a non-factor vector of character strings called `age.chars`.

CHAPTER 6

Data Frames

Thus far we've dealt with single lists of datum values, encoded in R as vectors. We are now ready to discuss a table of datum values, which is represented in R via a `data frame`. This chapter offers an introduction to data frames and some common operations with them.

6.1 Structure of A Data Frame

An Example

A famous example data frame that comes pre-loaded in R is a table of car performance data published by Motor Trend Magazine in 1974. If you type the variable name `mtcars` in the console you will see the following data:

(R.6.1)
```
                   mpg cyl  disp  hp drat    wt  qsec ...
Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46 ...
Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02 ...
Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61 ...
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44 ...
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02 ...
Valiant           18.1   6 225.0 105 2.76 3.460 20.22 ...
Duster 360        14.3   8 360.0 245 3.21 3.570 15.84 ...
Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00 ...
Merc 230          22.8   4 140.8  95 3.92 3.150 22.90 ...
...
```

Note that the color-highlighting we employ for `mtcars` is the same as variables we define. This is because `mtcars` IS a variable that somebody has defined in R – it's just that this variable is automatically loaded when R starts. Thus `mtcars` isn't a protected keyword in R. To convince yourself of this, note that there is no problem in issuing the command `mtcars <- 5`, which just rewrites the contents of the `mtcars` variable.

We won't really need it, but for more info on the meanings of the various columns, type `help(mtcars)`.

The dots ... indicate that there is more data that is not shown.

For each model of car, there is a measurement of miles per gallon, the # of cylinders in the engine, etc. The visual arrangement of the data in R mimics that of tables like Table 1.5.

Rows and Columns

A `row` of the data-frame (in agreement with the terminology in Sec. 1.3.1) is comprised of all the data in one horizontal line, as indicated in Fig. 6.1. The first row of `mtcars` is thus

(R.6.2)
```
Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46 ...
```

and the fourth row is

(R.6.3)
```
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44 ...
```

Each row in a data frame has a label. In the case of `mtcars`, the labels are the model of car. Thus, `Mazda RX4`, `Merc 240D` etc. are the labels for each row.

Figure 6.1: A *row* is a horizontal line of data in a data frame. Typically the data in a row are comprised of different types of measurements for some object/sample of interest.

Figure 6.2: A *column* is a vertical line of data in a data frame. Typically the data in a column are all measurements of the same type.

The color coding shown in `Mazda RX4` will be used exclusively for row and column labels within data-frames.

A `column` of the data-frame is comprised of all the data in one vertical line, as indicated in Fig. 6.2. The first column is thus

```
mpg
21.0
21.0
22.8
...
```
(R.6.4)

and the third column is

```
disp
160.0
160.0
108.0
...
```
(R.6.5)

Each column in a data frame has a label. In the case of `mtcars`, the column labels are the various measurements or properties of the cars, such as `mpg` and `disp`.

6.2 Data Selection in Data Frames

There is a package called `dplyr` that lets you select data frame rows and columns with a different syntax that some people find easier to read. However, the syntax covered here also works with things like vectors and matrices.

To select only a portion of the data in a data frame, we will use the same methods discussed in Sec. 5.1: bracket notation, and the `subset()` function. However, for a data frame we have to specify both the rows and the columns we want.

Selection With Bracket Notation

To select data in a data frame, such as `mtcars`, we can still use the square bracket notation `mtcars[ , ]`, but now there will be separate specification for rows (before the comma) and columns (after the comma). For example, to see the datum value in the 5th row and the 2nd column, which is the number of cylinders in the engine of the Hornet Sportabout, you could type `mtcars[5,2]`, as illustrated in Fig. 6.3 below.

Reminder that the row labels `Mazda RX4` etc. aren't considered a data column, so the first column is `mpg`.

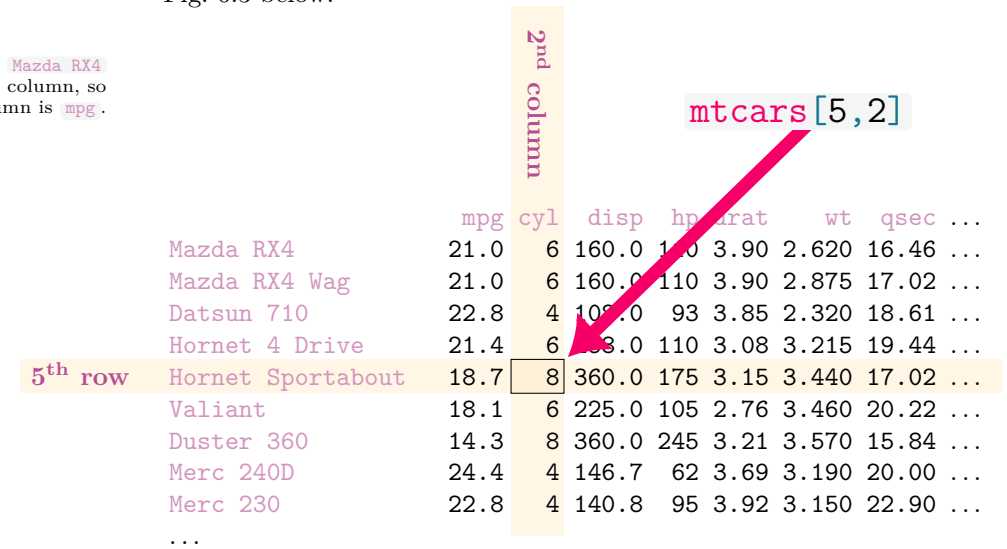


Figure 6.3: Location of the datum in row 5, column 2, which is selected via `mtcars[5,2]`.

Selecting data frame elements uses the same notation we saw in Table 5.5, with two modifications: you need to specify both rows and columns, and you can also refer to positions with row and column labels. Some examples are shown below in Table 6.1:

To get `mtcars` value(s) ...	type ...
in 1st row and 3rd column	`mtcars[1,3]` or `mtcars["Mazda RX4","disp"]` or `mtcars[1,"disp"]`
in 1st row and all columns	`mtcars[1,]` or `mtcars["Mazda RX4",]`
in rows 2 and 3 and columns 5 and 6	`mtcars[c(2,3),c(5,6)]` or `mtcars[c("Mazda RX4 Wag","Datsun 710"),c("drat","wt")]` or `mtcars[c(2,3),c("drat","wt")]`
in all rows and column 2	`mtcars[,2]` or `mtcars[,"cyl"]` or `mtcars$cyl`

Table 6.1: Various ways of selecting data frame (here `mtcars`) values based on location.

In the last row of Table 6.1 note that we can extract a single column of data by placing a dollar sign after the data frame name, and then writing the column label without quote marks, e.g. `mtcars$cyl`. There are thus three different ways to select a column of values:

The dollar-sign notation is compact and can make some expressions easier to read.

$$\texttt{mtcars[,1]} \quad = \quad \texttt{mtcars[,"mpg"]} \quad = \quad \texttt{mtcars\$mpg} \tag{6.1}$$

As was the case for vectors, we can also use conditional expressions to select for data frame elements based on datum values. Let's say we want the data only for rows in which `mpg` is greater than 20. We can insert `mtcars$mpg > 20` into the first slot of the bracket notation. The code

(R.6.6)
```
mtcars[mtcars$mpg>20,]
```

By virtue of Eq. (6.1), we can also get the same `mpg` data via `mtcars[mtcars[,1]>20,]` or even `mtcars[mtcars[,"mpg"]>20,]` though these are a little tougher to read.

returns a data frame with only rows where `mpg` is greater than 20, and with no restrictions on columns (i.e. all columns are present).

We could also place restrictions on the columns. To see only the `hp` and `cyl` data for cars with `mpg` greater than 20, we could type

(R.6.7)
```
mtcars[mtcars$mpg>20,c("hp","cyl")]
```

or equivalently

(R.6.8)
```
mtcars[mtcars$mpg>20,c(2,4)]
```

Table 6.2 below shows various examples of this syntax. To make the table more compact, we've renamed `mtcars` as `M` via `M <- mtcars`.

To get the following data from `M` ...	type ...

all `disp` data
for cars with
`cyl` =8
	`M[M$cyl==8,"disp"]`
	or
	`M[M$cyl==8,3]`
	or
	`M[M[,"cyl"]==8,"disp"]`
	or
	`M[M[,2]==8,3]`

all `mpg` data
for cars with
`hp` between 100 and 150
	`M[(M$hp>=100)&(M$hp<=150),"mpg"]`
	or
	`rows.wanted <- (M$hp>=100)&(M$hp<=150)`
	and then `M[rows.wanted,"mpg"]`
	or
	`M[(M[,"hp"]>=100)&(M[,"hp"]<=150),"mpg"]`
	or
	`M[(M[,4]>=100)&(M[,4]<=150),1]`

all `hp` and `mpg` data
for cars with
`cyl` =6
	`M[M$cyl==6,c("hp","mpg")]`
	or
	`M[M[,2]==6,c(4,1)]`

all data columns
for cars with
`cyl` =4 and `mpg` >25
	`M[(M$cyl==4)&(M$mpg>25),]`
	or
	`M[(M[,2]==4)&(M[,2]>25),]`

Table 6.2: More examples of selecting data frame values using bracket notation. We have first typed `M <- mtcars`.

Selection Using The `subset()` Function

Sometimes expressions like those in Table 6.2 can be cumbersome to both write and read. An alternative method of selecting data frame elements is to use the `subset()` function introduced in Sec. 5.1.2. The syntax is similar to that of Eq. (5.1), except that we now must specify both rows and columns:

$$\texttt{subset}\left(\boxed{\begin{array}{c}\text{name of}\\ \text{data frame}\end{array}}\,,\ \boxed{\begin{array}{c}\texttt{subset=}\\ \text{row values}\\ \text{you want}\end{array}}\,,\ \boxed{\begin{array}{c}\texttt{select=}\\ \text{columns}\\ \text{you want}\end{array}}\right) \tag{6.2}$$

To see all columns, omit the `select` parameter.

We specify the rows and columns the same way as with bracket notation, except instead of placing them inside brackets, we equate them to the `subset` and `select` parameters. For example, to reproduce the first row of Table 6.2, where we wanted `disp` data for cars with `cyl` = 8, we could type

```
subset(M, subset=cyl==8, select="disp")
```
(R.6.9)

Reminder that the spaces after the commas in (R.6.9) are optional, and are only used to make the expression easier to read.

and all its variations (you could enter `3` instead of `"disp"`, etc). If you prefer to leave out the parameter names you can just type

```
subset(M, cyl==8, "disp")
```
(R.6.10)

Example 6.1: Selecting Subset of Data Frame

Problem: Using the data from `M<-mtcars`, create a dataframe `x` containing `mpg`, `wt`, and `disp` data for cars with: `cyl` = 6, and `hp` between 100 and 200. Use both bracket notation and `subset()`.

Solution: To keep our R code easier to read our approach will be to separately define the rows and columns we want, and then insert these into both brackets and the `subset()` function.

Rows – We want rows that meet three conditions at once:

1st: `M$cyl==6`

2nd: `M$hp>100`

3rd: `M$hp<200`

Reminder that to express a condition of equality you need to use a double-equal sign `==`.

To also include possible values of `hp`=100 and `hp`=200, use the operators `>=` and `<=` instead of `>` and `<`. It won't matter for this particular dataset, since none of the cars have `hp` equal to exactly 100 or 200.

We can insist all three conditions are met at the same time by inserting the *and* operator `&` between each of them. We can thus define the rows we want via

```
rows.wanted <- (M$cyl==6)&(M$hp>100)&(M$hp<200)
```

Columns – We want the columns `mpg`, `wt`, and `disp`. We can define the columns we want via

```
columns.wanted <- c("mpg","wt","disp")
```

We can now insert these into bracket notation to define `x` via

```
x <- M[rows.wanted, columns.wanted]
```

or use the `subset()` function to define `x` via

```
x <- subset(M, subset=rows.wanted, select=columns.wanted)
```

or just

```
x <- subset(M, rows.wanted, columns.wanted)
```

with the resulting data shown in Fig. 6.4.

	mpg	wt	disp
Mazda RX4	21.0	2.620	160.0
Mazda RX4 Wag	21.0	2.875	160.0
Hornet 4 Drive	21.4	3.215	258.0
Valient	18.1	3.460	225.0
Merc 280	19.2	3.440	167.6
Merc 280C	17.8	3.440	167.6
Ferrari Dino	19.7	2.770	145.0

Figure 6.4: Data frame `x` resulting from selection in Example 6.1.

Note that we could have replaced our code above with the longer expression

```
x<-M[(M$cyl==6)&(M$hp>100)&(M$hp<200),c("mpg","wt","disp")]
```

and similarly with the `subset()` function.

Example 6.2: Fraction of Data Meeting Condition

Problem: What fraction of the cars in the `mtcars` data frame have a fuel efficiency of over 25 mpg?

Solution: To find the total number of cars (the number of rows), we can use the idea from Example 5.4. Because the `length()` function will count the number of entries in single vector, we can simply select any one column and find it's length. Let's pick `mpg`:

```
n.total <- length(mtcars$mpg)
```

which yields a value `n.total` =32 total cars. Alternatively, we can use the function `nrow()`:

```
n.total <- nrow(mtcars)
```

We can count the number of cars with `mpg` >25 by finding the length of the corresponding subset:

```
n.high.mpg <- length(mtcars[mtcars$mpg>25,"mpg"])
```

giving the value `n.high.mpg` =6. The total fraction of cars with `mpg` >25 is given by

```
n.high.mpg/n.total
```

which is $6/25 = 0.24$ or $\boxed{24\%}$.

Applying the `length()` function to an entire data frame simply counts the number of columns. The reason is that a data frame is actually what's known in R as a `list`, which is just a collection of things that can be of different types (numbers, vectors, strings, etc). A vector is a special type of list where all the entries are the same type. A data frame is really just a list vectors (the columns) of equal lengths, and the `length()` function returns the number of elements in a list. Thus, `length(mtcars)` returns 11, since there are 11 total columns of data, where each of these columns is itself a vector.

6.3 Creating a Data Frame

The `data.frame()` Function

Data frames are created via the `data.frame()` function. To see how it works, we'll put the data from Table 6.3 into a data frame.

student	sex	income (k$)	GPA
1	F	25	3.7
2	M	32	2.9
3	M	45	3.4
4	F	67	2.1
5	F	68	2.7
6	M	95	3.7
7	M	112	3.1

Table 6.3: Sample data table for a study of the relationship between sex, income, and GPA of college freshman. Same as Table 1.5.

We first create vectors for the data in the columns, remembering that vectors of categorical data are best stored as factors:

```
sex <- factor(c("F","M","M","F","F","M","M"))
income <- c(25,32,45,67,68,95,112)      #family income in k$
gpa <- c(3.7,2.9,3.4,2.1,2.7,3.7,3.1)   #out of 4.0
```

(R.6.11)

We don't need to put the student numbers into a column, because these are just labels for the rows and R will create these labels by default. We now create the data frame `college.data` via

```
college.data <- data.frame(sex,income,gpa)
```

(R.6.12)

You should be aware that even if we hadn't defined `sex` as a factor in (R.6.11), be default `sex` would be converted to a factor by the `data.frame()` function. If we didn't want this to happen, in (R.6.12) we could have used the parameter setting `stringsAsFactors=FALSE`.

Entering `college.data` at the console gives the output

(R.6.13)
```
  sex income gpa
1   F     25 3.7
2   M     32 2.9
3   M     45 3.4
4   F     67 2.1
5   F     68 2.7
6   M     95 3.7
7   M    112 3.1
```

Typing `summary(college.data)` returns level counts for the factor `sex` and 5 number summaries for the quantitative variables `income` and `gpa`.

Row and Column Labels

When creating the data frame, if you would like to assign column labels that are different from the names of the vectors (say `"S"`, `"I"`, and `"G"`), type

(R.6.14)
```
college.data <- data.frame("S"=sex, "I"=income, "G"=gpa)
```

though it's typically better to have slightly more informative column names.

To change the row labels to something other than numbers, we can use the `rownames()` function. For example, if we want to label the rows with the actual names of the (fictional) students, we could use the command

(R.6.15)
```
rownames(college.data) <- c("Mary B.","Adam K.","Cory L.",
     "Stephanie W.","Amy D.","Martin J.","Sandra H.")
```

and then typing `college.data` at the console gives

(R.6.16)
```
             sex income gpa
Mary B.        F     25 3.7
Adam K.        M     32 2.9
Cory L.        M     45 3.4
Stephanie W.   F     67 2.1
Amy D.         F     68 2.7
Martin J.      M     95 3.7
Sandra H.      M    112 3.1
```

You can also assign the row labels when you define the data frame. The following code would result in the same data frame as (R.6.16):

(R.6.17)
```
sex <- factor(c("F","M","M","F","F","M","M"))
income <- c(25,32,45,67,68,95,112)      #family income in k$
gpa <- c(3.7,2.9,3.4,2.1,2.7,3.7,3.1)   #out of 4.0
row.labels <- c("Mary B.","Adam K.","Cory L.",
     "Stephanie W.","Amy D.","Martin J.","Sandra H.")
college.data <- data.frame(row.names=row.labels,sex,income,gpa)
```

Finally, just as `rownames()` renames rows, you can use the same syntax with `colnames()` to rename columns.

Adding Rows and Columns

There are no special functions needed for deleting rows or columns, since we can just create a new data frame using the selection rules discussed in Sec. 6.2. However, one handy bit of syntax is that if you specify a row or column with a *negative* number, you will get everything *except* that row or column. Thus, `mtcars[-2,]` returns a data frame without the 2nd row. The command `mtcars[,c(-3,-4)]` returns a data frame without the 3rd and 4th columns.

To add a column of data to a data frame, we can use the `cbind()` function. For example, imagine we wanted to add age data to the table. We could type

(R.6.18)
```
age <- c(23,19,35,43,22,20,27)    #age in years
college.data <- cbind(college.data,age)
```

yielding the data frame

```
              sex income gpa age
Mary B.         F     25 3.7  23
Adam K.         M     32 2.9  19
Cory L.         M     45 3.4  35
Stephanie W.    F     67 2.1  43
Amy D.          F     68 2.7  22
Martin J.       M     95 3.7  20
Sandra H.       M    112 3.1  27
```
(R.6.19)

To add a row, we can create a new data frame with just a single row via

```
new.row <- data.frame(row.names="Julius P.",sex="M",income=18,
                      gpa=3.5,age=28)
```
(R.6.20)

Make sure the new one-row data frame has the *exact* same column labels as the original data frame. Also make sure the value of any factor variables match the level definitions. For example, if we accidentally typed a lowercase "m" as `sex="m"` then this would be added to the data frame as a new level.

We can then add this data frame to `college.data` using the `rbind()` function:

```
college.data <- rbind(college.data,new.row)
```
(R.6.21)

resulting in the data frame

```
              sex income gpa age
Mary B.         F     25 3.7  23
Adam K.         M     32 2.9  19
Cory L.         M     45 3.4  35
Stephanie W.    F     67 2.1  43
Amy D.          F     68 2.7  22
Martin J.       M     95 3.7  20
Sandra H.       M    112 3.1  27
Julius P.       M     18 3.5  28
```
(R.6.22)

6.4 Importing & Exporting Data

Often you'll be importing data from outside R. This data can be stored in different ways in different types of files. Accordingly, R has a separate import function for different types of files. The standard file type is what's known as a `comma separated values file` or a `.csv file`, which is a plain text file where each line of the file is a row of data, with values separated by commas.

To see what a .csv file looks like, we can use the `write.csv()` function to export the data frame `college.data` shown in (R.6.22) to a .csv file:

```
write.csv(college.data,"collegeData.csv")
```
(R.6.23)

```
"","sex","income","gpa","age"
"Mary B.","F",25,3.7,23
"Adam K.","M",32,2.9,19
"Cory L.","M",45,3.4,35
"Stephanie W.","F",67,2.1,43
"Amy D.","F",68,2.7,22
"Martin J.","M",95,3.7,20
"Sandra H.","M",112,3.1,27
"Julius K.","M",18,3.5,28
```

Figure 6.5: Contents of the .csv file created by (R.6.23).

This will create a plain text file (in your R working directory) with the contents shown in Fig. 6.5. This .csv file is in a form that can be used by a wide variety of software.

You can import data from a .csv file using the `read.csv()` function. For example, you could create a new data frame `duplicate` by importing the .csv file we just made:

(R.6.24)
```
duplicate <- read.csv("collegeData.csv",row.names=1)
```

There are quite a few different parameters one can specify for `read.csv()` in order to deal with the exact formatting details of the .csv file or the resulting data frame. Use `help(read.csv)` and `help(write.csv)` for more info.

where the `row.names` parameter indicates which column in the .csv file has the row names. If you don't specify the `row.names` parameter then R will just assume you want to label the rows with numbers. At any rate, the data frame `duplicate` will be identical to the data frame `college.data`.

If you have data stored in some other format, for example in a spreadsheet format, or the data format of another statistical program (like SAS or SPSS), it's typically best to try and use those programs to save the data as a `.csv` file. If you don't have the ability to do that, R has plenty of functions that can import other file types. For some examples, visit the DataCamp Quick-R website [7].

6.5 Sorting

Sometimes we wish to have a data frame with a particular column of data in ascending or descending order. To do this, in the row slot we use the `order()` function on the column we wish to sort by. For example, to see `college.data` from (R.6.22) in order of age, you can type

(R.6.25)
```
college.data[order(college.data$age),]
```

resulting in the data frame being sorted in order of ascending age, as shown in Fig. 6.6.

	sex	income	gpa	age
Adam K.	M	32	2.9	19
Martin J.	M	95	3.7	20
Amy D.	F	68	2.7	22
Mary B.	F	25	3.7	23
Sandra H.	M	112	3.1	27
Julius P.	M	18	3.5	28
Cory L.	M	45	3.4	35
Stephanie W.	F	67	2.1	43

Figure 6.6: `college.data` sorted by `age` in ascending order.

To sort in order of descending age, put a minus sign in front of the sort column inside the `order()` function. OR USE DECREASING=FALSE Thus, the command

(R.6.26)
```
college.data[order(-college.data$age),]
```

gives the output shown in Fig. 6.7.

	sex	income	gpa	age
Stephanie W.	F	67	2.1	43
Cory L.	M	45	3.4	35
Julius P.	M	18	3.5	28
Sandra H.	M	112	3.1	27
Mary B.	F	25	3.7	23
Amy D.	F	68	2.7	22
Martin J.	M	95	3.7	20
Adam K.	M	32	2.9	19

Figure 6.7: `college.data` sorted by `age` in descending order.

You can also sort by a factor column. The command

(R.6.27)
```
college.data[order(-college.data$sex),]
```

will sort the data frame by sex. By default the levels will appear in alphabetical order, i.e. "F" rows first and then "M". If you wish them to appear in a different order, then you can define the print order of the levels, as shown in (R.5.17). If we redefine the `sex` column as

(R.6.28)
```
college.data$sex <- factor(college.data$sex,levels=c("M","F"))
```

then (R.6.27) would result in "M" rows appearing first, followed by "F" rows.

6.6 Exercises

1. Find a dataset with at least 10 samples (rows) and at least 3 data columns. This can be data you've collected in a project, data from a journal article, data from somebody's research here at SBC, data from the EPA website, etc. It can be from anywhere (just not "made up" or built into R).

   Make a labeled data frame in R containing the data. The data can be either numerical or qualitative. Use comments in the R script to indicate the meaning and physical units (e.g. "rows are individual birds", "column 1 is length in ft", "column 2 is mass in kg", etc.).

   i) Use `rbind()` to add a new row of data to the data frame (you can just make up some data).

   ii) Use `cbind()` to add a new column of data to the data frame (you can just make up some data).

   iii) Give the command to select the element in row 4, column 2.

   iv) Give the command to select rows 2 through 7.

   v) Store the first column of the data frame to a vector named `x`, using the selection syntax discussed in Sec. 6.2.

   vi) Store the third column of data frame to a vector named `y`.

   vii) Use `write.csv()` to export your data frame to a .csv file.

   viii) Use `read.csv()` to read the .csv file you just made and store it in a new data frame.

2. Use the selection syntax of Sec.6.2 to select the following data from `mtcars`:

   i) `hp`, `cyl`, and `mpg` data for cars that have at least 150 hp.

   ii) `gear` and `drat` data for cars that have between 17 and 23 `mpg`.

   iii) `mpg` and `drat` data for 6-cylinder cars that have between 17 and 23 `mpg`.

Submit your R code, and cite the source of your dataset in a comment in the R file.

CHAPTER 7

More About Data Frames

In this chapter we discuss four aspects of data frames that will prove useful as we move forward: making use of factor encoding, simplifying code via the `with()` function, summarizing quantitative data with the `pairs()` function, and aggregating data with `aggregate()`.

7.1 Making Use of Factor Encoding in Graphs

There are many applications of factors in R. One of them is making nicer plots. Consider again the `college.data` data frame defined in (R.6.17). We could make a plot of `gpa` vs `income` as follows:

(R.7.1)
```
plot(x=college.data$income,
     y=college.data$gpa,
     main="GPA vs Family Income",
     xlab="Family Income (in k$)",
     ylab="GPA (out of 4.0)",
     pch=19)
```

where `pch` is explained in Fig. 7.1. The resulting plot is shown in Fig. 7.2.

We can use factors to color-code the data points, so that `"F"` and `"M"` points are different colors. Doing so requires putting together three different ideas.

Idea #1 - Manually Coloring the Points

We can manually color each of our 7 data points by setting the `col` parameter of `plot()` equal to a vector of 7 colors. Modifying (R.7.1) as follows will plot female data points blue, and male data points red:

(R.7.2)
```
plot(x=college.data$income,
     y=college.data$gpa,
     main="GPA vs Family Income",
     xlab="Family Income (in k$)",
     ylab="GPA (out of 4.0)",
     pch=19,
     col=c("blue","red","red","blue","blue","red","red"))
```

with result shown in Fig. 7.3. Manually specifying the colors for a large dataset would be very time consuming, so we want a more efficient method of creating this vector of colors.

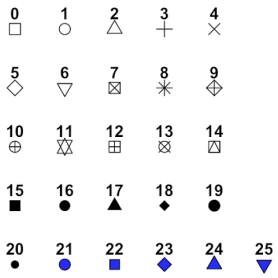


Figure 7.1: Some common choices for the `pch` parameter indicating the shape of the plot points. You can control the *size* of these shapes with the `cex` parameter, where `cex=1` is 100% size.

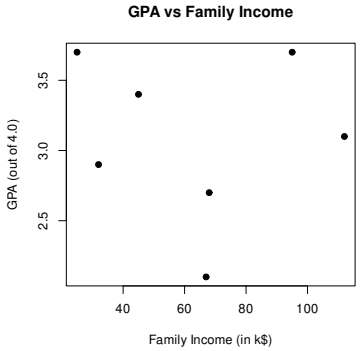


Figure 7.2: Scatterplot of GPA vs Income for `college.data`.

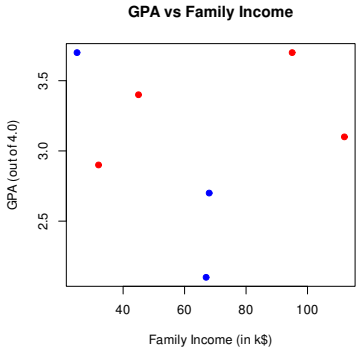


Figure 7.3: Scatterplot of GPA vs Income for `college.data` with female data points blue and male datapoints red.

Idea #2 - Creating A Vector of Color Values

Consider a vector

```
x <- c("blue","red")
```
(R.7.3)

We have seen that `x[1]` returns `"blue"` and `x[2]` returns `"red"`. However, recall from Sec. 5.1.1 that we can also put a vector of locations in the brackets, including repeats. For example,

```
x[c(1,2,2,2,1)]
```
(R.7.4)

will return a vector containg the the 1st, 2nd, 2nd, 2nd, and 1st datum value in `x`, i.e. `"blue" "red" "red" "red" "blue"`. Using this idea, we could create the vector of 7 color values in (R.7.2) with the code

```
c("blue","red")[c(1,2,2,1,1,2,2)]
```
(R.7.5)

though by itself this is still quite inefficient for a large dataset.

Type `colors()` to see a list of the 657 built-in colors in R. A quick web search provides visuals for the different R color names. You can also specify custom colors using a hex code, e.g. `col="#32A852"`. There are many websites that help generate hex codes.

Idea #3 – The `unclass()` Function

The final piece of the puzzle is the `unclass()` function discussed in Sec. 5.3. If you feed a factor to this function, it will return a vector of the integer level values of the factor data. Thus, the code

```
unclass(college.data$sex)
```
(R.7.6)

will return the vector `1 2 2 1 1 2 2`, which is the exact vector we typed in manually in (R.7.5)! Hence, we can automatically generate the color coding by changing line 7 of (R.7.2) to:

```
7       col=c("blue","red")[unclass(college.data$sex)]
```
(R.7.7)

This method works quickly for any number data points.

As a finishing touch, we can add a grid and a legend to make the plot a bit nicer. The complete plot code is:

```
1  plot(x=college.data$income,
2       y=college.data$gpa,
3       main="GPA vs Family Income",
4       xlab="Family Income (in k$)",
5       ylab="GPA (out of 4.0)",
6       pch=19,
7       col=c("blue","red")[unclass(college.data$sex)],
8       panel.first=grid())
9  legend(x=62.5,y=3.6,
10        legend=c("Female","Male"),
11        pch=19,
12        col=c("blue","red"))
```
(R.7.8)

A few notes on (R.7.8). In line 8, the `panel.first` parameter is used so that the gridlines are drawn first, and the points are then drawn on top of the grid. Also, the closing parenthesis on this line closes out the `plot(` function on line 1. On line 9 the `x` and `y` position refer to the upper left corner of the legend box, and some trial and error is usually required for a nice result.

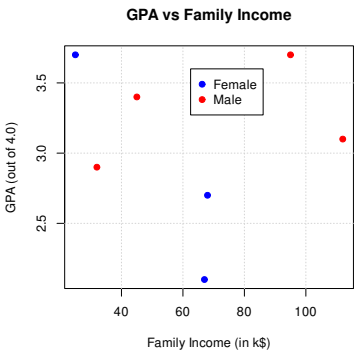


Figure 7.4: Scatterplot of GPA vs Income for `college.data`.

The result is shown in Fig. 7.4.

Example 7.1: Color Coding Data Points in `iris`

Problem: The pre-loaded dataset `iris` contains measurements of sepal and petal dimensions for three different species of Iris - *Iris setosa*, *Iris versicolor*, and *Iris viginica*. Make a plot of petal length vs. sepal length, with color coding for the three different species.

The command `help(iris)` gives more info on this well-known dataset.

Solution: Issuing the command `head(iris)` or `str(iris)` indicates the relevant structure of the data frame. In particular, the relevant column labels are `Sepal.Length`, `Petal.Length`, and `Species`.

The command `levels(iris$Species)` indicates that the factor levels are, in order, `"setosa" "versicolor" "virginica"`. Let's say we want these data points to be red, blue, and green. Following (R.7.7), we can generate a list of colors via

```
c("red","blue","green")[unclass(iris$Species)]
```

As a matter of taste, we think the plot will look better with open dots `pch=21` and colored fills or backgrounds, meaning we'll use the `bg` parameter instead of `col` (which would only color the outline of the circle). The rest of the plot code is similar to (R.7.8):

```
1  plot(x=iris$Sepal.Length,
2       y=iris$Petal.Length,
3       main="Iris Petal vs. Sepal Length",
4       xlab="Sepal Length (cm)",
5       ylab="Petal Length (cm)",
6       pch=21,
7       bg=c("red","blue","green")[unclass(iris$Species)],
8       panel.first=grid())
9  legend(x=6.7,y=2.8,
10         legend=c("setosa","versicolor","virginica"),
11         pch=21,
12         pt.bg=c("red","green","blue"))
```

with the result shown in Fig. 7.5.

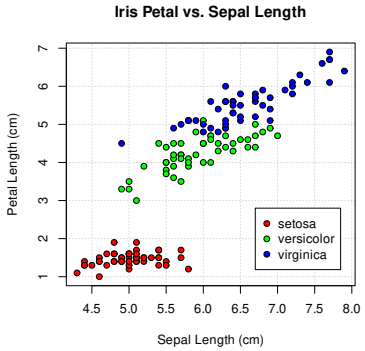


Figure 7.5: Color-coded scatter plot of petal vs. sepal length for `iris` data.

On line 12 in the legend specification you need to use the parameter `pt.bg` to fill the point colors. Using just `bg` will simply change the color of the entire legend background.

7.2 The `with()` Function

Having to prefix every column with the data frame name and the dollar-sign, as in `iris$Sepal.Length`, can sometimes create R code that is cumbersome to read. A simple option is to to use the `with()` function, in which you specify a parent data frame and enclose a sequence of commands in braces. For example, the plot code from Example 7.1 can be rewritten as

Sometimes you'll see code (outside this book) that uses `attach()` and `detach()` to perform a function that appears similar to `with()`. There are some traps and pitfuls in using `attach()`, and it is generally advised to not use it.

(R.7.9)

```
1  with(iris,{
2     plot(x=Sepal.Length,
3          y=Petal.Length,
4          main="Iris Petal vs. Sepal Length",
5          xlab="Sepal Length (cm)",
6          ylab="Petal Length (cm)",
7          pch=21,
8          bg=c("red","blue","green")[unclass(Species)],
9          panel.first=grid())
10    legend(x=6.7,y=2.8,
11           legend=c("setosa","versicolor","virginica"),
12           pch=21,
13           pt.bg=c("red","green","blue"))
14 })
```

The braces `{` and `}` on lines 1 and 14 are needed if you perform more than one R command inside `with()`.

In line numbers 2, 3, and 8, notice the column labels without the prefix of `iris$`. For this particular code there isn't too much of a difference, but sometimes `with()` can really help code readability. We'll make use of this function in (R.7.17) later in the chapter.

7.3 The `pairs()` Function

In data frames with multiple columns of quantitative data, we can quickly see how the various columns relate to each other with the `pairs()` function. Using the `iris` data frame as an example, where the first 4 columns are quantitative, the code

```
pairs(iris[,1:4])
```

(R.7.10)

produces the plot shown in Fig. 7.6 below.

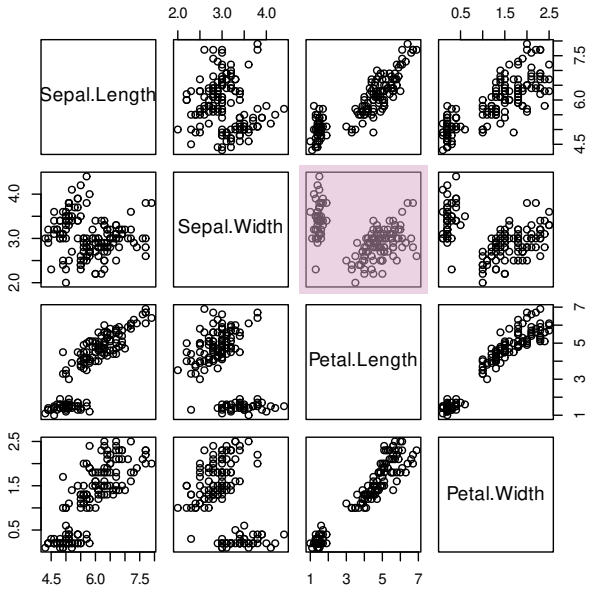


Figure 7.6: Pairs-plot for quantitative data in `iris`. The highlighted square has `Sepal.Width` on the y-axis and `Petal.Length` on the x-axis.

For a given row in the plot, the y-axis is the data for the label in that row, and the x-axes are the data for the labels in the corresponding column.

We can use the methods outlined in Sec. 7.1 to make the plot nicer. The code

(R.7.11)
```
pairs(iris[1:4],
      main="Iris Data",
      pch=21,
      bg=c("red","blue","green")[unclass(iris$Species)])
```

produces Fig. 7.7.

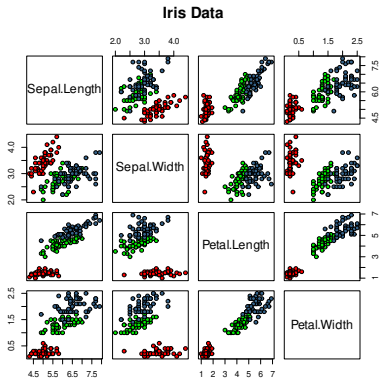


Figure 7.7: `iris` data pairs plot color-coded by species.

7.4 The aggregate() Function

Often we want to average over certain quantities in a data frame. One convenient way to do this in R is to use the `aggregate()` function. To see this in action, we'll consider the built-in `ChickWeight` data frame.

Issuing `ChickWeight` and `help(ChickWeight)` gives us some info about this data frame. 50 `Chick`s are raised on 4 different `Diet`s, and body `weight`s in grams are taken at `Time`s of birth, every other day until day 20, and also day 21. We would like to make a graph of weight versus time, and color code the different diets. The following code makes use of the techniques discussed in this chapter:

(R.7.12)
```
1  DietColors <- c("red","blue","green","orange")
2  plot(x=ChickWeight$Time,
3       y=ChickWeight$weight,
4       xlab="Time (days)",
5       ylab="Weight (grams)",
6       main="Chick Growth From Different Diets",
7       pch=19,
8       cex=0.25,
9       col=DietColors[unclass(ChickWeight$Diet)],
10      panel.first=grid())
11 legend(x=3,
12        y=340,
13        title="Diet",
14        legend=c("Red","Green","Blue","Orange"),
15        pch=19,
16        col=DietColors)
```

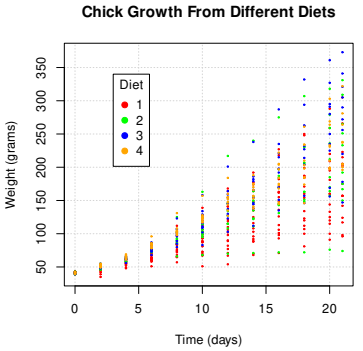


Figure 7.8: Plot produced by (R.7.12). This plot is hard to read.

The result, shown in Fig. 7.8, is hard to read. One problem is that there are many points clustered together. We can artificially spread the points out a bit by using the `jitter()` function. If we replace line 2 above with

(R.7.13)
```
2  plot(x=jitter(ChickWeight$Time),
```

we get the plot in Fig. 7.9. This plot is still tough to read, however, because there are many different chicks on each diet. For example, to find the chickens on diet #3, we can use the `unique()` function, which returns all the unique values in a vector. The code

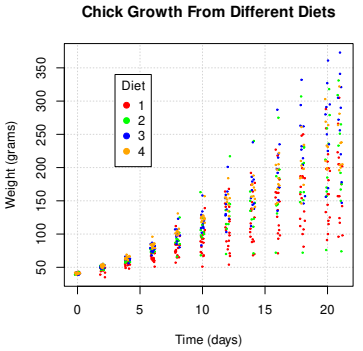


Figure 7.9: Plot produced by (R.7.13). The use of `jitter()` is a slight improvement, but it's still hard to interpret the data.

(R.7.14)
```
unique(ChickWeight[ChickWeight$Diet==3,"Chick"])
```

returns `31 32 33 34 35 36 37 38 39 40`, meaning there are 10 different chickens on diet #3. Hence, in Fig.7.9, for each `Time` there are 10 different blue dots, and similarly for the other colors.

The graph would be much easier to read if we only plotted the *mean* weight of all the chicks on each diet. To do this quickly, we can use `aggregate()`, which requires 3 pieces of information:

We wish to combine all of the chicks on each diet for each time, and summarize the results with a mean. Combining and summarizing data in this way is known as `aggregating` data, hence the name of the R runction.

`aggregate` (`x =` name of data frame , `by =` list of columns to group data by , `FUN =` function to apply to other columns)

We are interested in finding the mean weight at each combination of `Diet` and `Time`. In other words, we want to *group* the data by `Diet` and `Time`. We therefore specify the following:

`x =` name of data frame = `ChickWeight`

`by =` list of columns to group data by = `list(ChickWeight$Diet,ChickWeight$Time)`

`FUN =` function to apply to other columns = `mean`

Technically the `by` parameter specifies a list of *factors* to group the data by. The resulting data frame will have values grouped by the factor levels of the specified columns. Non-factor columns will be temporarily coerced into factors for the purpose of the aggregation.

We will store the resulting data frame in the variable `CW` as

(R.7.15)
```
CW.by <- list("Diet"=ChickWeight$Diet,
              "Time"=ChickWeight$Time)
CW <- aggregate(x=ChickWeight, by=CW.by, FUN=mean)[,1:3]
```

Note that R will give you a warning message for the code in (R.7.15), because we are trying to apply the `mean()` function to columns like `Diet` which contain qualitative data. There is nothing to worry about, though. R will just put `NA` in these columns.

We kept only the first 3 columns: the two grouping factors, and the weights. For each combination of `Diet` and `Time`, the weights have been averaged over all of the `Chick`s. Using `CW` in the plot code of (R.7.12) (with a few tweaks)

(R.7.16)
```
DietColors <- c("red","blue","green","orange")
plot(x=CW$Time,
     y=CW$weight,
     xlab="Time (days)",
     ylab="Weight (grams)",
     main="Chick Growth From Different Diets",
     pch=19,
     cex=0.7,
     col=DietColors[unclass(CW$Diet)],
     panel.first=grid())
legend(x=3,
       y=240,
       title="Diet",
       legend=c("Red","Green","Blue","Orange"),
       pch=19,
       col=DietColors)
```

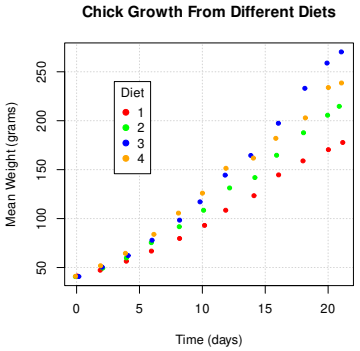


Figure 7.10: Plot produced by (R.7.16) with weights averaged over the chicks on each diet via `aggregate()`.

results in Fig. 7.10, which is much easier to read.

We could make the plot even nicer by adding `lines()` through the points. One way to do this is to subset the data by `Diet` and use 4 different `lines()` commands. We'll make use of the `with()` function to simplify the code. Appending the following code to the end of (R.7.16)

(R.7.17)
```
20 with(subset(CW,Diet==1),
21      lines(x=Time,y=weight,col=DietColors[1]))
22 with(subset(CW,Diet==2),
23      lines(x=Time,y=weight,col=DietColors[2]))
24 with(subset(cW,Diet==3),
25      lines(x=Time,y=weight,col=DietColors[3]))
26 with(subset(CW,Diet==4),
27      lines(x=Time,y=weight,col=DietColors[4]))
```

results in Fig. 7.11, which does a nice job of visually summarizing the data.

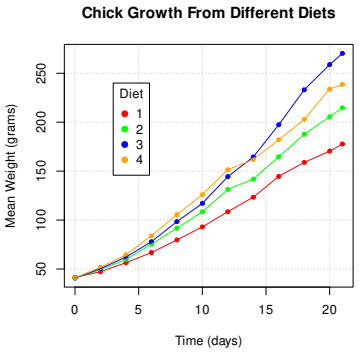


Figure 7.11: Plot produced by (R.7.16) and (R.7.17).

Comment 7.1 – Using for Loops (Optional Reading)

In (R.7.17) the same command is repeated 4 times, the only difference being a different number 1 thru 4 is used. A better way to write such code is to use the `for()` command. In place of (R.7.17) we could write

```
22 for (i in 1:4)
23 {
24         with(subset(cw,Diet==i),
25              lines(x=Time,y=weight,col=DietColors[i]))
26 }
```

Line 22 tells R to repeat the commands that follow for values of `i` equal to 1, then 2, then 3, and then 4. Any commands inside the braces `{` and `}` will be repeated. Programmers call such a repetition a *loop*.

This approach saves a lot of time and effort as the number of repetitions increases. Also, if we want to change the code that's being repeated, we only have to change it once. This also makes typing errors less likely, and makes the code easier to read.

There are other ways to implement loops in R. For example, functions such as `apply()` and `lapply()` are very convenient ways to perform general repetitions on data frames and lists. If you want to improve your skills in R, it would be time well spent to research these methods online and practice with them!

Example 7.2: Aggregate iris Data By Species

Problem: Use `aggregate()` to summarize the mean values of the `iris` dataset measurements (discussed in Ex. 7.1) for each species of iris.

Solution: We only wish to find means for the first 4 columns, since the 5th column, `Species`, is qualitiative. Hence, we will specify a data frame of `x=iris[,1:4]`.

We wish to group the mean lengths by species, hence we'll set

`by=list("Species"=iris$Species)`

Finally, we wish to apply the `mean` function to the other columns, so we set `FUN=mean`. The code

Reminder that for pre-loaded datasets like `iris` you can use the `help()` function to find some info about the data, i.e. type `help(iris)` at the console.

Using the entire data frame `iris` instead of `iris[,1:4]` will result in `aggregate()` trying to find the mean of the qualitative data in the `Species` column, giving a final column of `NA` values in the resulting data frame. This is nothing to worry about, though.

Omitting the `"Species"=` in line 2 just means that the first column of the resulting data frame will have the label `Group.1` instead of `Species`.

```
1 aggregate(x=iris[,1:4],
2           by=list("Species"=iris$Species),
3           FUN=mean)
```

produces the data frame

Species	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
setosa	5.006	3.428	1.462	0.246
versicolor	5.936	2.770	4.260	1.326
virginica	6.588	2.974	5.552	2.026

Thus, the mean sepal length for the *Iris setosa* data is 5.006 cm, etc. We see that *Iris virginica* seems to be the largest of the the three, though interestingly *Iris setosa* has the largest sepal width despite its other 3 measurements being the smallest.

7.5 Exercises

1. The pre-loaded data frame `Loblolly` contains measurements of height vs. time for 14 different Loblolly pine trees (each from a different seed). Use `aggregate()` to create a data frame `lob.mean` containing the mean height (averaged over all trees) for each time. Use `lob.mean` to make a fully labelled plot of mean height vs. time, and add `lines()` to the plot.

2. The pre-loaded data frame `CO2` contains data related to the cold tolerance of the grass *Echinochloa crus-galli*.

   i) Use `aggregate()` to create a data frame `CO2.mean.uptake` containing CO2 uptake rates averaged over `Plant`s, with data grouped by `conc`, `Type`, and `Treatment`. Depending upon how you did things, you may end up with columns of duplicate data. Make sure your final data frame has only columns for conc, uptake, type, and treatment.

   ii) For only the *nonchilled* plants in `CO2.mean.uptake`, use the techniques of Sec. 7.1 to make a fully labelled scatterplot of mean CO2 uptake rates vs. ambient CO2 concentration, color-coded by origin (red = Mississippi, blue = Quebec). Use solid dots (`pch=19`) for the plot points.

   iii) Now add averaged uptake points to your plot for the *chilled* plants. You can do this by using the function `points()` instead of `plot()`. Keep the same color-coding, but now make the plot points open-circles via `pch=21`.

   iv) Add a legend to the plot indicating the meaning of the different points. Note that you can specify a vector of `pch` values.

   v) According to the plot, the plants in which region seem to suffer a smaller drop in CO2 uptake when the temperature drops? In other words, between Quebec and Mississippi, which plants seem to tolerate the cold better?

3. The pre-loaded data frame `airquality` contains daily air quality measurements for New York in 1973. Make a `pairs()` plot for this data. Based on this plot, what appears to be the relationship between ozone and each of the other measurements? For example, how does ozone seem to relate to solar radiation? Does ozone increase with increasing radiaiton, or does it decrease? Or is there not really a clear relationship? Etc.

CHAPTER 8

Limitations of Descriptive Statistics

Typically the purpose of using statistics is to infer or conclude something about the world we live in. Descriptive statistics simply restates or describes collected data, and since data is always in some sense incomplete, descriptive statistics by itself is insufficient for *inferring* things about the world. In this chapter we motivate the idea that in order to conduct proper *statistical inference*, we will need to make use of the theory of probability.

8.1 Measurement Noise

In considering the impact of noise on statistical inference, we will first look at the pre-loaded dataset `InsectSprays` and try to infer from the data which of the sprays is most effective.

Figure 8.1: A tomato hornworm, the pesticide target in `InsectSprays`.

Photo by Amanda Hill, licensed under CC0, downloaded from Wikimedia.org.

8.1.1 Descriptive Statistics

`InsectSprays` contains measurements of *Manduca quinquemaculata* (tomato hornworm) counts from 72 plots of land. The 72 plots were divided into 6 groups of 12, and each group was treated with a different pesticide (labelled A thru F). Since counting *every* insect on a plot of land is not feasible, random sampling was used (i.e. counts were made for randomly chosen smaller quadrats within each plot).

We can get a sense of the structure of the data frame by looking at the first and last six rows using `head()` and `tail()`. Issuing the command `head(InsectSprays)` yields the data frame shown in Fig. 8.2 and the command `tail(InsectSprays)` yields the data frame shown in Fig. 8.3.

You can get a quick idea of the detailed structure of any R variable via the `str()` function. The command `str(InsectSprays)` quickly lets you know there are 72 data points with two variables - `count` contains numerical data, and `spray` contains factor (qualitative) data with 6 levels. And as usual, you can find more background info via `help(InsectSprays)`.

	count	spray
1	10	A
2	7	A
3	20	A
4	14	A
5	14	A
6	12	A

Figure 8.2: First six rows of `InsectSprays` via `head()`.

One quick way to investigate the data is to look at side-by-side box plots with `boxplot.()` We can specify a data frame with the `data` parameter, and give the two columns of interest (here `count` and `spray`) separated by a tilde `~`. The code

(R.8.1)
```
1 boxplot(count~spray,
2         data=InsectSprays,
3         main="Insect Counts By Spray",
4         xlab="Spray",
5         ylab="Insect Counts")
6 abline(h=seq(0,25,5),lty="dotted",col="gray")
7 boxplot(count~spray,add=TRUE)
```

produces the plot shown in Fig. 8.4. Line 6 adds some horizontal grid lines on top

	count	spray
67	13	F
68	10	F
69	26	F
70	26	F
71	24	F
72	13	F

Figure 8.3: Last six rows of `InsectSprays` via `tail()`.

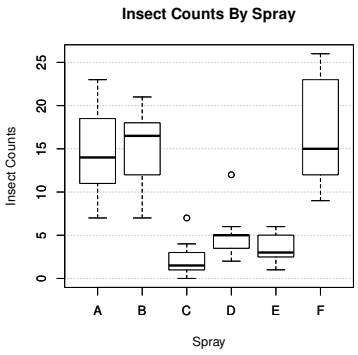


Figure 8.4: Box plot of the data in `InsectSprays`.

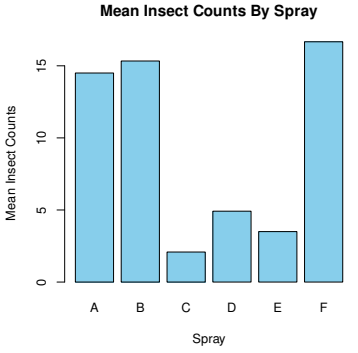


Figure 8.5: Bar plot of mean count data in (R.8.3). Note that if you would like to have horizontal grid lines behind the bars, you can use the same technique shown in lines 6 and 7 of (R.8.1).

of the graph, and line 7 replots the boxes so that they are on top of the gridlines. Looking at the IQR for the sprays, we see there is a fair amount of spread in the count data. Still, we can get a quick visual of how the spray counts compare with each other.

The middle bars of each box in Fig. 8.4 indicate the median of the count data for each spray. If we wish to compare the mean of the counts for each spray, one approach is to use `aggregate()`. The code

```
IS.mean <- aggregate(x=InsectSprays,
                     by=list("spray"=InsectSprays$spray),
                     FUN=mean)[,1:2]
```
(R.8.2)

results in the data frame

```
  spray     count
1     A 14.500000
2     B 15.333333
3     C  2.083333
4     D  4.916667
5     E  3.500000
6     F 16.666667
```
(R.8.3)

We can graphically summarize the information in (R.8.3) using a barplot. Using the same tilde `~` and `data` notation as in (R.8.1),

```
barplot(count~spray,
        data=IS.mean,
        main="Mean Insect Counts By Spray",
        xlab="Spray",
        ylab="Mean Insect Counts",
        col="skyblue")
```
(R.8.4)

we get Fig. 8.5.

8.1.2 So Which Spray Is The Best?

The descriptive statistics of the data shows that spray C had smaller average insect counts than the other sprays. But does this mean that, in general, spray C is more effective at repelling/killing insects than the others? Maybe not. The reason is that there are *many* things influencing the insect counts, not just the pesticide. Some other factors that could influence insect counts for the plots might include:

- the temperature, humidity, sunlight, etc. near the plants
- the quality of the plants (better tomatos or peppers or tobacco leaves or whatever maybe attract more bugs)
- variations in nearby insect predators/parasites (either of the hornworms or the adult moths who lay eggs)
- uneven spreading of the pesticide
- variations in the soil composition

- measurement error in counting bugs (maybe somebody missed one, or double counted)
- variation due to the random sampling (maybe the quadrat chosen wasn't typical compared to the rest of the plot)

and so on. No matter how well one controls a scientific experiment, there will always be a bunch of factors that will influence the measurements. In other words, there is a lot of potential variation in the insect counts that is due to things other than the pesticide.

To visualize this potential variation in counts, let's consider pesticide E, which had the 2nd fewest average counts. Let us suppose we could define a number $N_{\text{E,true}}$ as the "true" number of insects one would count for pesticide E if *all* possible influencing factors were exactly their average values in all the plots of land. Let $N_{\text{E,measured}}$ be the number of counts actually recorded in a quadrat of a plot.

$$N_{\text{E,true}} = \text{counts expected if everything is exactly average}$$
$$N_{\text{E,measured}} = \text{counts actually measured}$$

If all variables are exactly at their average value, then by definition $N_{\text{E,measured}}$ should equal $N_{\text{E,true}}$. We represent this on the number line in Fig. 8.6 below.

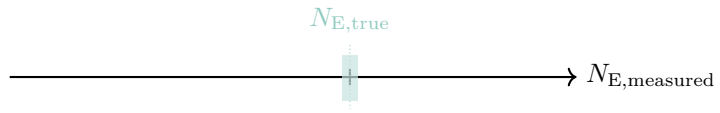


Figure 8.6: $N_{\text{E,measured}}$ should equal $N_{\text{E,true}}$ if all possible influencing factors are exactly their average values.

Let's now imagine that the temperature is a bit warmer than normal, favoring the presence of hornworms. This would cause $N_{\text{E,measured}}$ to be a bit larger than $N_{\text{E,true}}$, as shown below in Fig.8.7.

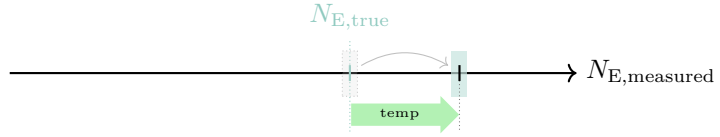


Figure 8.7: The measured counts $N_{\text{E,measured}}$ will be slightly larger than the "true" count $N_{\text{E,true}}$ if the temperature is higher than usual (either due to mean ambient temperature, or an increased amount of sunlight), favoring the presence of the hornworms.

However, imagine that there are also a lot more birds near this field, for whatever reason. This will result in more caterpillars being eaten by birds than is typical. Hence, this variable would result in $N_{\text{E,measured}}$ going down a bit:

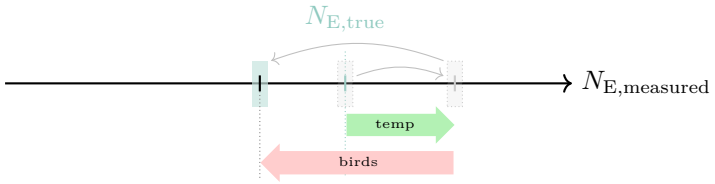


Figure 8.8: The measured counts $N_{\text{E,measured}}$ will be reduced due to the presence of extra birds.

Perhaps the counts are also decreased because the plant/fruit quality is less than average:

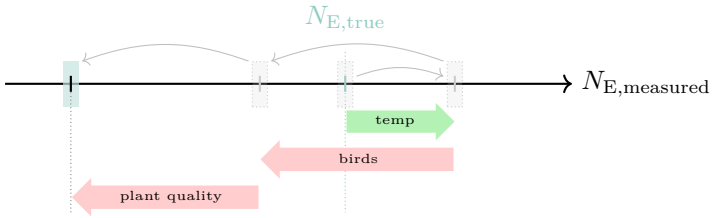


Figure 8.9: $N_{\text{E,measured}}$ would go down if the plant quality in the sampled quadrat were less than average.

But at the same time, counts might be enhanced due to a poor coverage of the pesticide spray, and so on for every possible variable:

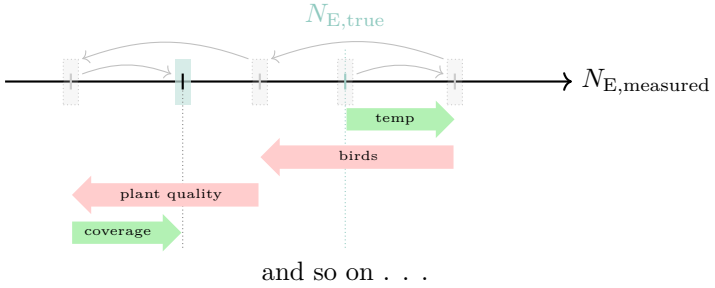


Figure 8.10: $N_{\text{E,measured}}$ would go up if the coverage of the pesticide was worse than average. And so it goes for a large number of different variables, as $N_{\text{E,measured}}$ bounces around on the number line, eventually landing at its final value. All of these variations in $N_{\text{E,measured}}$ result in *noise* in the data.

It is clear that at the end of all this, we are very unlikely to have $N_{\text{E,measured}}$ be exactly equal to $N_{\text{E,true}}$. There is just too much possible variation in the counts due to variables we aren't measuring, which is the definition of `noise`, as discussed back in Sec. 0.2. The noise in the count data is due to our ignorance of (or uncertainty around) the unmeasured variables mentioned above.

What this all means is that pesticide E might actually be more effective (on average) than pesticide C, but in the 12 quadrats we measured the mean $N_{\text{E,measured}}$ turned

out to be larger than $N_{\mathrm{E,true}}$, just due to all those other random variables we didn't measure. In other words, C might have had the fewest counts just by luck. Even though the data seems to suggest that C is the best, we can't be certain.

8.1.3 The Need For Probability

At the end of the day, even if we take more data, do more experiments, try to be more careful, etc., the fundamental fact remains that *any inferences we make will have some level of uncertainty*, due to our inability to measure and account for all the various factors mentioned above. The best we will be able to do is say things like

> We believe that C is better than E, but there is a 23% chance we got "unlucky" and that E is actually better.

We will also want to infer *how much* better C is. For example, if C is 40% better than E, that's a big deal, but if C is only 0.01% better than E, nobody will care. We will also need probability to quantify our uncertainty in how much better one spray is compared to another.

But where would the number 23% come from? The branch of mathematics that deals with quantifying uncertainty in this way is *probability*. In statistics, in order to move past descriptions of the data and *infer* some conclusion about the world, we will need to use the theory of probability.

8.2 Uncertainties in Regression

As discussed in Sec. 0.2, regression analysis involves studying how various variables or measurements relate to each other mathematically. Let's consider the pre-loaded dataset `airquality`, containing air quality measurements for New York City back in 1973. The first four columns contain the measurements.

The code

(R.8.5)
```
pairs(airquality[1:4],
      main="New York City Air Quality Data, 1973")
```

yields Fig. 8.11. Examining the plot, we see that some of the variables don't seem to have a pronounced relationship with each other. For example, the relationship between solar radiation `Solar.R` and `Wind` isn't all that clear. On the other hand, there does seem to be a noticeable relationship between temperature `Temp` and `Ozone`.

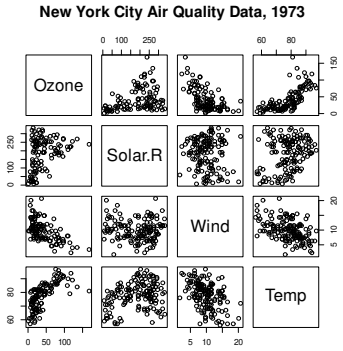


Figure 8.11: `pairs()` plot from (R.8.5).

We can make a plot of temperature vs. ozone with the code

(R.8.6)
```
1 plot(x=airquality$Ozone,
2      y=airquality$Temp,
3      main="Temperature vs. Ozone",
4      xlab="Ozone (ppb)",
5      ylab=expression(paste("Temperature ",degree,"F")),
6      pch=19,
7      cex=0.7,
8      panel.first=grid())
```

with result in Fig. 8.12.

In line 5 of (R.8.6), we have used the `paste()` function to assemble multiple strings of characters into a single string. The `expression()` function on the outside converts the keyword `degree` to the degree symbol. Of course we could just use `ylab="Temp F"`, which makes the code simpler, but the degree symbol makes the plot look more professional. The `expression()` function can be used to typeset a variety of mathematical expressions in this way. See `help(plotmath)` for more examples.

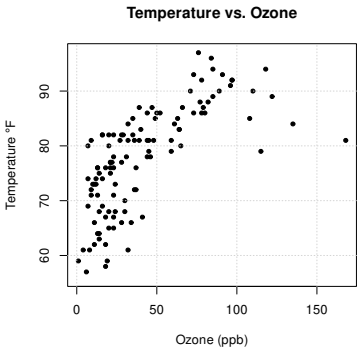


Figure 8.12: Plot from (R.8.6).

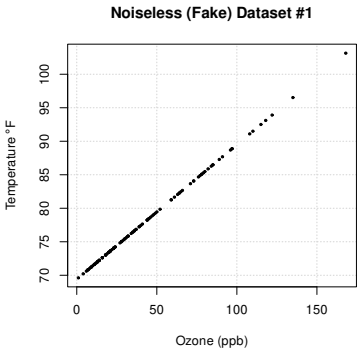


Figure 8.13: A made-up, idealized dataset where there is no noise in the data, and where temperature varies linearly with ozone.

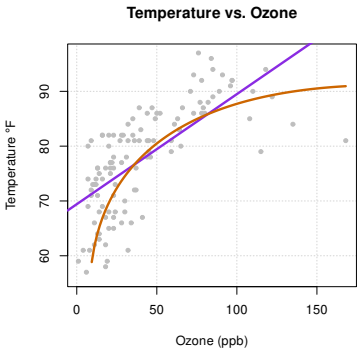


Figure 8.14: The actual data with two possible "noiseless" regression curves superposed.

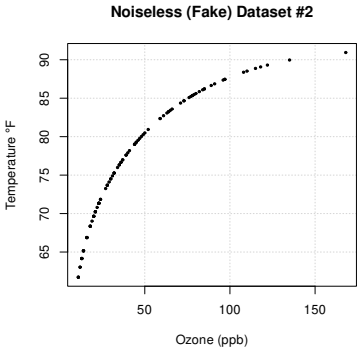


Figure 8.15: A made up, idealized dataset where there is no noise in the data, and where temperature varies with ozone according to the curve in Fig. 8.14.

The plots indicate that, rougly speaking, higher ozone values correspond to higher temperatures, though it would appear that the noise in the data is substantial. A regression analysis would attempt to find the exact mathematical relationship between the variables assuming noise was negligible (i.e. assuming all other variables we're not measuring are exactly their "average" values).

There will always be some level of noise in data. But as a thought experiment (this could never actually happen), let's imagine that all of our ozone data was totally without noise, and that the temperatures measured were exactly accurate. Perhaps in this case the temperature depends on ozone linearly. Then the data might look something like shown in Fig. 8.13. If the data really looked like this, we could simply pick any two data points and use basic algebra to find the equation of the line going through them. For the fake data in Fig. 8.13, it turns out this equation would be approximately

$$\texttt{Temp} = 0.2008 \times \texttt{Ozone} + 69.41 \tag{8.1}$$

meaning in Fig. 8.13 the slope of the line is 0.2008 and the y-intercept is 69.41.

Unfortunately, it is *very* unlikely that Eq. 8.1 is accurate, for two primary reasons: one, we have no idea if the relationship is actually linear, and two, the data *does* have a lot of noise, meaning there is no reason to believe the line in Fig. 8.13 is accurate. Let's examine these two issues in more detail.

8.2.1 Uncertainty in Form of Regression Equation

In Fig. 8.13 we simply assumed that the "noiseless" relationship between temperature and ozone is linear. But is this assumption valid? Perhaps if we had an incredibly detailed understanding of atmospheric physics we might predict a linear relationship on the basis of theory, but that's not the case here. Furthermore, our original data in Fig. 8.12 doesn't makes it obvious that the "noiseless" relationship is linear.

In Fig. 8.14, we have shown two possible "noiseless" relationships superposed over the original data. It's not really clear that the straight line fits the data better than the curve. In other words, maybe if we were able to erase the noise from the data, instead of Fig. 8.13 we would have Fig. 8.15.

To make matters worse, in addition to the two curves in Fig. 8.14, there are in fact *many* different curves that might best describe the relationship between temperature and ozone. So how do we know which type of regression equation (or type of curve) is the right one? As you might have guessed, we'll never know the answer with *certainty*, and the best we can do is inform the analysis with theory and more data, and then at the end quantify our uncertainty via probability theory.

8.2.2 Uncertainty in Values of Regression Parameters

For the sake of argument, let's assume that the correct type of regression equation is a straight line. We still have the problem that the noise in the data prevents us from knowing with certainty *which* straight line is correct. In Fig. 8.16, two possible regression lines for the unknowable "noiseless" data are superposed on the

original data. Given all the noise, how do we know which one is correct? In other words, how do we really know the value of the slope and y-intercept?

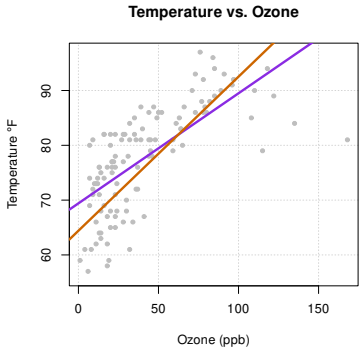


Figure 8.16: Two possible regression lines fitting the "noiseless" data. How do we know which one is correct?

As before, we can never know the slope and y-intercept with certainty. The best we can do is give our best mathematical guesses, and quantify the uncertainty. We might say something like:

> "We are 68% certain that
>
> - the slope is between 0.18 and 0.22
> - the intercept is between 68.39 and 70.44
>
> assuming that there is a linear relationship."

where these numbers are computed with probability theory. For the record, it's not necessarily clear what is precisely meant by a statement like "we are 68% certain that . . .". We will discuss this issue in detail in Part II of the book.

8.3 Description vs. Inference

For a given situation of interest, if we somehow had *all* possible data and information, then there would be no uncertainty in anything, and descriptive statistics would tell the whole story. In reality, we never have complete information, and we need to *go beyond* the descriptive statistics in order to *infer* some fact about the world. This process of going beyond the data itself in order to draw conclusions about the world is called `statistical inference`.

Our analysis of the `InsectSprays` and `airquality` datasets suggests that in order to perform proper statistical inference, we will need to make use of the theory of probability. This will be the main subject of Part II of the book.

8.4 Exercises

1. Consider the pre-loaded dataset `ToothGrowth`.

   i) Explain the structure and contents of this data frame. What do the rows and columns mean? How many observations are there? What do you think is the purpose of the experiment? Etc.

   ii) Following (R.8.1), make a labelled boxplot of `len` vs. `supp`, but only for `dose` equal to 2.0 mg/day. Now do the same for doses of 1.0 and 0.5.

   iii) Based on the boxplots, which supplement type has the higher median tooth length?

   iv) Use `aggregate()` to compute mean tooth lengths for each combination of supplement and dose.

   v) Which supplement has the higher mean tooth length?

If you're feeling fancy, you can actually have all three boxplots show up in the same image if you issue the command `par(mfrow=c(1,3))` before you issue the boxplot commands. This command arranges all subsequent plots in a 1×3 grid. This works for all plot types.

See (R.7.15) for an example of how to use `aggregate()` in this way.

vi) Notice that as dose increases, the median/mean length for the supplement types get closer together. What do you make of this fact?

vii) What are possible sources of noise in the length data?

viii) Based on your analysis of `ToothGrowth`, which supplement type is generally more effective at facilitating tooth growth? Explain why you cannot be certain in your conclusion.

2. Consider the pre-loaded dataset `iris`, which was examined in Ex. 7.1.

i) Make a labelled plot of petal length vs. sepal length for *Iris versicolor*.

ii) What are some possible sources of noise for the data?

iii) Assuming somehow that the data was "noiseless", what would be your guess as to the type of regression curve relating the two variables? Explain why you cannot be certain in your guess.

You can use `help(abline)` for info on how to do this. Make sure you read about the meaning of different possible parameters, such as `h`, `v`, `a`, and `b`. You can also consult the internet for guidance if you are stuck.

iv) Let's assume that the regression curve is linear. Using `abline()`, on top of the plot data draw your best guess for the exact regression line. What are its slope and y-intercept? Explain why you cannot be certain that these values are accurate.

Part II

Probability

CHAPTER 9

Introduction to Probability

In this chapter we introduce the concept of probability, discussing some of the basic underlying ideas and also introducing some notation. In the next chapter we will discuss the basic rules of probability that will prove useful in more complicated calculations.

9.1 Ranking Plausibility

It is a part of everyday thinking to associate an amount of plausibility with facts or explanations. If you see somebody sip liquid out of a cup and yell "ahhhhh!" and wince in pain, you may think *"they probably drank liquid that was too hot"*. You're not certain about your hot-coffee theory, but the explanation fits the observed facts pretty well. Another possible explanation might be *"the exact moment they took their first sip, a bad muscle cramp happened"*. However, this seems less probable or likely than our first explanation, given that it would be a somewhat unusual coincidence.

In general, intuitively we tend to *rank* the plausibility of explanations or occurrences. Some explanations or hypotheses seem more plausible than others.

Example 9.1: Rank Plausibility of Hypotheses

Problem: Rank the plausibility of the following hypotheses regarding the extinction of dinosaurs:

A) Aliens visited earth and killed them all.

B) A massive volcanic erruption killed them all.

C) A meteor struck the Earth and the resulting climate changes killed off all large land animals.

D) Every dinosaur on Earth had a heart attack at the exact same time and dropped dead.

Solution: In the author's opinion, hypothesis D is basically impossible. A coincidence like that would just never happen. Hypothesis A seems very very implausible, as it would require not only that aliens exist, but also that they somehow were capable of killing every dinosaur on Earth. Perhaps not impossible, but close to it. Between B and C, evidence seems to suggest that C is more likely than B.

So the author's ranking would be C, B, A, D, from most to least plausible.

We can also rank the plausibility of outcomes of some event.

Example 9.2: Rank Plausibility of Possible Outcomes

Problem: Jennifer intends on rolling a fair 20-sided die (numbers 1 thru 20 on the sides). Rank the plausibility that she will:

A) roll a 5

B) roll an odd number

C) roll a 21

Solution: Let's assume the die is fair (i.e. it isn't weighted in such a way that one particular side is favored), and also that she's not cheating.

Outcome C is impossible, since there are only numbers 1 thru 20 on the sides. E has no plausibility at all.

Outcome A is possible, but rolling a single specific number isn't all that plausible.

Outcome B is the most plausible of all, since half of the sides are odd numbers (1,3,5,..). On average, C happens half the time the die is rolled.

Hence, from most to least plausible, the ranking is B, A, C.

9.2 Probability = Quantified Plausibility

Probability theory involves assigning numbers to our plausibilities. Something that is definitely true or certain is given the number 1. Something that is definitely false or impossible is given the number of 0. In this book we will use the symbol $\mathbb{P}$ to denote the probability of an event or explanation.

Most other books denote probabilities with a regular P or p instead of $\mathbb{P}$.

Let A, B, C, etc. be items to which we assign plausibility. These items could be events that supposedly happened, or they could be events that may happen in the future, or they could be explanations for observed facts, etc. We use the following notation:

$$\mathbb{P}(A) \equiv \text{ the probability of } A \tag{9.1}$$

Reminder that $\equiv$ means "is defined to be equal to".

to indicate the quantitative measure from 0 to 1 of how plausible/certain you find A to be. The probability of the other items is then $\mathbb{P}(B)$, $\mathbb{P}(C)$, etc.

9.3 Probability Depends On What You Know

Probabilities are inherently subjective, meaning they depend upon our state of knowledge in the world. This is easily illustrated with a few examples:

Example 9.3: Subjectivity of Probability

Problem: Nola flips a fair coin, gets a heads (denoted H), and looks at the result. Meanwhile, Allen is in another room and can't see the result. What does each person assign to $\mathbb{P}(H)$ right after the flip?

Solution: Nola is looking at the result, which is heads. Thus, from her point of view, it is 100% certain or plausible that she flipped a heads. Hence, she would say $\mathbb{P}(H) = 1$.

Allen doesn't have access to the result, however. So for him, either H or T is equally plausible. So he would say $\mathbb{P}(H) = 0.5$.

These two probabilities are different, because Nola and Allen have different information available.

If Nola says "ooh - a heads!" and Allen hears her, then he now has more information or knowledge. If he knows Nola is trustworthy, he would now say $\mathbb{P}(H) = 1$. If he knows she might be lying about it to trick him, he wouldn't think $\mathbb{P}(H) = 1$. Again, probabilities change as our state of knowledge changes.

Example 9.4: Subjectivity of Probability II

Problem: Nola and Allen are told that Paul is about to attempt a 3-point shot in a basketball game. Nola is told that Paul is a professional basketball player, but Allen is told that Paul is a 6-year old child. Who will estimate a higher probability of Paul making the shot?

Solution: Nola knows that NBA players make 3 pointers around 35% of the time on average, so she might estimate $\mathbb{P}(\text{makes a 3-pt shot}) = 0.35$.

Allen knows that most 6-year-olds would have a hard time even getting the ball near the basket from that distance, so he would estimate a low probability, perhaps $\mathbb{P}(\text{makes a 3-pt shot}) = 0.005$.

Because Nola and Allen have access to different information, they arrive at different probability estimates. Whatever the specific numbers, clearly Nola will estimate the higher probability.

Probabilities are always dependent upon the information avaialable. In other words, probabilities are *conditional*, depending upon what info we have.

The word *condition* here refers to some particular circumstance. In example Ex. 9.3 above, if the condition (or circumstance) is that Paul is a pro player, then $\mathbb{P}$ has one value. If the condition (or circumstance) is that Paul is a child, then $\mathbb{P}$ will have a different value.

Let A represent an event or hypothesis or explanation for which we wish to compute a probability. Let B represent all the information we have (including events we assume have happened and hypotheses we assume are true). We then use the following notation:

$$\mathbb{P}(A|B) \equiv \mathbb{P}(A) \text{ given that B is 100\% certain or true or known} \tag{9.2}$$

A probability written this way is known as a `conditional probability`. The following are 4 different ways of saying $\mathbb{P}(A|B)$ out loud:

- "the probability of A conditioned on B"
- "the probability of A conditional on B"
- "the probability of A assuming B"
- "the probability of A given B"

Example 9.5: Conditional Probability of Making A 3-Pointer

Problem: Restate the conclusions of Ex. 9.3 using the conditional probability notation of Eq. (9.2).

Solution: Nola, believing Paul to be a professional player, would write her estimate as

$$\mathbb{P}\Big(\text{makes a 3 pt shot} \;\Big|\; \text{professional}\Big) = 0.35$$

while Allen, believing Paul to be a 6-year old, would write his estimate as

$$\mathbb{P}\Big(\text{makes a 3 pt shot} \;\Big|\; \text{6 year old}\Big) = 0.005$$

Because probabilities are conditioned upon available information, a person will often change or *update* their probability estimates as they learn new information.

Example 9.6: Updating Probabilities

Problem: John has a butterfly in his greenhouse. On Monday, John observes the morphology (size, shape, coloration, etc.) of the butterfly. Looking it up, he realizes this species is sexually monomorphic, meaning males and females look alike.

On Tuesday, John sees the butterfly lay eggs.

For each day, what is his estimate of the probability that the butterfly is a male?

Solution: Let $\mathbb{P}(M)$ denote the probability that the butterfly is male.

On Monday, John has no information to indicate whether the butterfly is M or F, since he just observes it flapping around, and both sexes look the same. Hence, M and F are equally plausible, and he would say that given only the morphology,

$$\text{Monday:} \quad \mathbb{P}\Big(M \;\Big|\; \text{observed morphology}\Big) = 0.5$$

On Tuesday, John has new information. Seeing that the butterfly layed eggs, he is now 100% certain that the butterfly is female. Hence, his updated probability is

$$\text{Tuesday:} \quad \mathbb{P}\Big(M \;\Big|\; \text{observed morphology \& layed eggs}\Big) = 0$$

Updating probabilites as we acquire new knowledge is a fundamental activity in science.

In situations where the conditions (known info) are obvious or well-understood, we often omit them from the notation. For example, if we wish to denote the

probability of flipping a coin and getting heads $\mathbb{P}(H)$, there are a number of implicit conditions. We assume the coin is fair, that it's being flipped by a person who isn't manipulating the flip, that it's taking place on Earth (and not outer space), that the coin has two sides, and so on. There are in fact a near infinite number of unspoken conditions. However, they're all obvious, so we don't bother writing all of them and we simply write

$$\mathbb{P}(H)$$

We will often write probabilities in this abbreviated way, though behind the scenes they are always conditional, depending upon what we know. The notation of Eq. (9.1) is itself written in this shortand way.

9.4 The Complement Rule

Let's say that A is a possible outcome for some event. It must be true that

$$\mathbb{P}(\text{not } A) = 1 - \mathbb{P}(A) \tag{9.3}$$

or equivalently

$$\mathbb{P}(A) + \mathbb{P}(\text{not } A) = 1 \tag{9.4}$$

This rule essentially says "we are completely certain that *some* outcome will occur", since (A) and (not A) cover all possibilities.

For example, if $\mathbb{P}(A) = 0.6$, this means we are 60% certain that A will happen. There is thus a 40% chance that A will *not* happen.

Example 9.7: The Complement Rule

Problem: A meteorologist says there is a 25% chance it will rain tomorrow. What is the probability it will *not* rain tomorrow?

Solution: According to the complement rule in Eq. (9.3),

$$\begin{aligned}\mathbb{P}(\text{no rain}) &= 1 - \mathbb{P}(\text{rain}) \\ &= 1 - 0.25 \\ &= 0.75\end{aligned}$$

Hence there is a 75% probability of no rain tomorrow.

9.5 The Principle of Insufficient Reason

An idea we'll use frequently is the `principle of insufficient reason`, which is often called `the indifference principle`. This principle states that if there

is no reason to suspect that A is favored over B or vice versa, then we should assign equal probabilities to both. If there are more than 2 outcomes/hypotheses, again the probabilities should all be equal.

We have actually used this principle several times already without explicitly saying so. For example, when flipping a fair coin, we have stated an outcome of heads H and an outcome of tails are equally plausible or probable. Let's use the variable x to denote this probability. In other words, let $\mathbb{P}(H) = \mathbb{P}(T) = x$. If we now use the complement rule, Eq. (9.3) tells us that

$$\begin{aligned}
\mathbb{P}(H) &= 1 - \mathbb{P}(\text{not } H) \\
\mathbb{P}(H) &= 1 - \mathbb{P}(T) && \leftarrow \text{used (not } H) = (T) \\
x &= 1 - x && \leftarrow \text{used } \mathbb{P}(T) = \mathbb{P}(H) = x \\
2x &= 1 && \leftarrow \text{added } x \text{ to both sides} \\
x &= \frac{1}{2} = 0.5 && \leftarrow \text{divided both sides by 2}
\end{aligned}$$

Thus, for a fair coin we have $x = \mathbb{P}(H) = \mathbb{P}(T) = 0.5$, which we just assumed in Ex. 9.3. One way of thinking about this result is that the total probability of 1 is split evenly between the two equivalent outcomes H and T.

9.6 Exercises

1. You take a break, and when you come back to your desk you see that your sandwich is missing. Come up with at least 5 explanations for its disappearance, and rank them by how plausible they are.

2. Nola and Allen are presented with samples of water, and they are going to be testing the E. coli levels. If there are more than 100 colonies per mL, the water will be classified as unsafe for drinking. Less than 100 colonies, and it is listed as safe (by WHO standards).

   Nola is told the water is from an isolated stream high up on a mountain that is remote from any human population, and Allen is told the water is from a river downstream from a sewage plant in a big city.

   Prior to running the tests, who will predict the higher probability that the water is safe to drink? What numbers would you estimate if you were in their position? Write your answers using conditional probability notation.

3. In a certain mammal, there is a 53% chance that a birth will be female. What is the probability a birth will be male? What assumptions did you make in arriving at your answer?

4. In a certain mammal, there is a 10% chance of having twins. Does this mean there is a 90% chance of having a single baby? Expain your reasoning.

5. You draw a card from a shuffled deck of playing cards.

   i) Which is more plausible - that you draw an ace, or a 2? Explain your reasoning.

   ii) Which is more plausible - that you draw an ace, or any numbered card? Explain.

CHAPTER 10

The Product and Sum Rules

In this chapter we introduce two fundamental quantitative rules that govern how we perform probability calculations: the product rule (used in *and* situations) and the sum rule (used in *or* situations).

10.1 The Product Rule (and)

The product rules states that

$$\mathbb{P}(A \text{ and } B) = \mathbb{P}(A) \cdot \mathbb{P}(B|A) \tag{10.1}$$

Note that $\mathbb{P}(A \text{ and } B)$ is often abbreviated as $\mathbb{P}(AB)$.

This means that the probability of both A and B together is the probability of A times the probability of B given that A happened (or is true). Whenever we see **and**, we use the product rule.

Example 10.1: Product Rule - Stumbles and Falls

Problem: During his morning walk to work, there is a 2% chance that Allen stumbles at some point. Furthermore, there is a 10% chance that Allen will fall whenever he stumbles.

What is the probability Allen stumbles and falls on the way to work one morning?

Always convert any percentages (by dividing by 100) before doing calculations. Here, 2% = 2/100 = 0.02 and 10% = 0.1. You can convert the final answer to a percent (by multiplying by 100) if you like.

Solution: We want $\mathbb{P}\Big(\text{stumbles } \textsc{and} \text{ falls}\Big)$, which is an *and* question.

According to the product rule in Eq. (10.1),

$$\begin{aligned}\mathbb{P}\Big(\text{stumbles } \textsc{and} \text{ falls}\Big) &= \underbrace{\mathbb{P}\Big(\text{stumbles}\Big)}_{=2\%} \times \underbrace{\mathbb{P}\Big(\text{falls}\Big|\text{stumbles}\Big)}_{=10\%} \\ &= 0.02 \times 0.1 \\ &= \boxed{0.002}\end{aligned}$$

A bit of logic: *the chance he has fallen given that he has stumbled* is **not** generally the same number as *the chance he has stumbled given that he has fallen*. In equation form, $\mathbb{P}(A|B) \neq \mathbb{P}(B|A)$. It is worthwhile to think through why these are different.

The probability that Allen stumbles and falls one morning is thus 0.002, or equivalently $0.002 \times 100 = 0.2\%$. That's not a big probability, but if Allen walks to work enough times, it's likely to happen at some point.

Note that we need to write *stumbles* and *falls* in the order indicated. While it's true $\mathbb{P}$(falls and stumbles) is *logically* the same thing as $\mathbb{P}$(stumbles and falls), we can't compute it directly using Eq. (10.1), because we are not given values either for $\mathbb{P}$(falls) or for $\mathbb{P}$(stumbles|falls).

At first glance an expression like $\mathbb{P}$(stumbles|falls) may seem strange, since it seems to denote the probability of stumbling *after* you've already fallen down, which is the wrong time order. But that is *NOT* the meaning of $\mathbb{P}(A|B)$. All that is denoted by $\mathbb{P}(A|B)$ is the probability of A given that B also occurs, and there is no specification of which comes first, or if one causes the other.

Example 10.2: Product Rule - Water Testing

Problem: Assume an E coli. test for a water sample is 98% accurate if there is at least 100 mL of water sample, and is only 80% accurate if there is less water than this. Mike is sort of sloppy when preparing samples, and there is a 15% chance one of his water samples will contain less than 100 mL.

What is the probability that Mike will properly prepare a sample with at least 100 mL of water, and that the test is nonetheless inaccurate?

Solution: This is an *and* question, where we need to compute

$$\mathbb{P}\Big(\geq 100 \text{ AND inaccurate}\Big)$$

and according to Eq. (10.1), we have

$$\mathbb{P}\Big(\geq 100 \text{ AND inaccurate}\Big) = \mathbb{P}\Big(\geq 100\Big) \times \mathbb{P}\Big(\text{innacurate}\Big| \geq 100\Big)$$

The probabilities on the right can be computed via the complement rule, Eq. 9.3.

$$\begin{aligned}\mathbb{P}(\geq 100) &= 1 - \mathbb{P}(\text{not } \geq 100)\\ &= 1 - \mathbb{P}(< 100)\\ &= 1 - 0.15\\ &= 0.85\end{aligned}$$

where the 0.15 was given in the problem, and

$$\begin{aligned}\mathbb{P}(\text{inaccurate}| \geq 100) &= 1 - \mathbb{P}(\text{accurate}| \geq 100)\\ &= 1 - 0.98 = 0.02\end{aligned}$$

where the 0.98 was given in the problem. Thus

$$\begin{aligned}\mathbb{P}\Big(\geq 100 \text{ AND inaccurate}\Big) &= \underbrace{\mathbb{P}\Big(\geq 100\Big)}_{=0.85} \times \underbrace{\mathbb{P}\Big(\text{innacurate}\Big| \geq 100\Big)}_{=0.02}\\ &= 0.85 \times 0.02\\ &= \boxed{0.017}\end{aligned}$$

so that there is a 1.7% chance that Mike gets the sample right *and* that the test is nonetheless inaccurate.

Note that we didn't use the given information that $\mathbb{P}(\text{accurate}| \leq 100) = 0.80$, since we were not asked about what happens if Mike's sample is *less* than 100 mL.

10.2 Independent Events

Note that in Ex. 10.1 and 10.2 the specified events were not independent, i.e. they were `dependent events`. For example, in Ex. 10.2, the accuracy of the test depends on whether the sample is at least 100 mL.

Often we wish to compute *and* probabilities for events that don't depend on each other in any way (meaning one can't influence the other). Let's assume A and B don't depend upon each other in any way. Remember that $\mathbb{P}(A|B)$ means "the probability of A assuming B is true", but if A is independent of B, then the

probability of A is the same whether or not B is true (or happened). So instead of $\mathbb{P}(A|B)$, we can just write $\mathbb{P}(A)$.

Thus, when A and B don't depend on each other we have

$$\mathbb{P}(A|B) = \mathbb{P}(A) \qquad \text{and} \qquad \mathbb{P}(B|A) = \mathbb{P}(B) \tag{10.2}$$

and these events are called `independent events`. For independent events, the product rule Eq. (10.1) simplifies to

$$\underbrace{\mathbb{P}(A \text{ and } B) = \mathbb{P}(A) \cdot \mathbb{P}(B)}_{\text{only if } A \text{ and } B \text{ are independent}} \tag{10.3}$$

Example 10.3: Flipping A Coin Twice

Problem: What is the probability of flipping a fair coin twice and getting two heads?

Solution: When you pull a coin from your pocket and flip it, the coin has "no memory" of what's happened before. Your probability of flipping a heads doesn't depend in any way on the result of some previous flip. This is true whether the previous flip was last month or just a minute ago.

Hence, consecutive coin flips are independent events, and we can use Eq. (10.3):

$$\mathbb{P}(HH) = \mathbb{P}(H) \cdot \mathbb{P}(H) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1 \cdot 1}{2 \cdot 2} = \frac{1}{4} = \boxed{0.25}$$

There, for a fair coin is thus a 25% probability of flipping two heads in a row.

A separate issue is whether or not there is really a 50% chance of getting heads on a flip. If the coin is slightly imbalanced, or if the tossing method slightly favors one side, then the number may be different from 50%. Let's say the probability of heads is actually 55%. Every time you go to toss the coin, this probability of heads remains 55%, no matter what the previous flips are. Consecutive flips will still be independent. The fairness of the coin/flips and the independence of the flips are two separate issues.

Reminder that $\mathbb{P}(AB)$ is shorthand for $\mathbb{P}(A \text{ and } B)$. Thus, $\mathbb{P}(HH)$ is just shorthand for $\mathbb{P}(H \text{ and } H)$.

The product rule for independent events extends in a straightforward way for more than two events. If we know that A, B, C etc. are independent, then

$$\underbrace{\mathbb{P}(A \text{ and } B \text{ and } C...) = \mathbb{P}(ABC...) = \mathbb{P}(A) \cdot \mathbb{P}(B) \cdot \mathbb{P}(C) \ldots}_{\text{only if } A,B,C... \text{ are independent}} \tag{10.4}$$

10.3 The Sum Rule (or)

10.3.1 General Form of Sum Rule

When dealing with *and* questions, we use the product rule. When dealing with *or* questions, we use the sum rule. The sum rule states that

$$\mathbb{P}(A \text{ or } B) = \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \text{ and } B) \tag{10.5}$$

We subtract the last term $\mathbb{P}(A \text{ and } B)$ so that we don't overcount if A and B can happen at the same time.

10.3.2 Sum Rule For Mutually Exclusive Events

Most of the time we use the sum rule, we'll be dealing with events that cannot both happen (or be true). For example, you can't flip both a heads and tails at the same time. Such events are called `mutually exclusive`, since A being true excludes the possibility that B is true, and vice versa. For mutually exclusive events, by definition $\mathbb{P}(A \text{ and } B) = 0$, and the sum rule takes the form

$$\underbrace{\mathbb{P}(A \text{ or } B) = \mathbb{P}(A) + \mathbb{P}(B)}_{\text{only if } A \text{ and } B \text{ are mutually exclusive}} \tag{10.6}$$

If there are more than 2 possibilities, just add them up: $\mathbb{P}(A \text{ or } B \text{ or } C) = \mathbb{P}(A) + \mathbb{P}(B) + \mathbb{P}(C)$, etc.

Example 10.4: Drawing a Queen Of Clubs

Problem: What is the probability of drawing a queen of clubs from a shuffled deck of 52 playing cards (no jokers)?

Solution: According to the complement rule of Eq.(9.4),

$$\mathbb{P}(Q\clubsuit) + \mathbb{P}(\text{not } Q\clubsuit) = 1 \tag{10.7}$$

According to the sum rule of Eq. (10.6), $\mathbb{P}(\text{not } Q\clubsuit)$ can be computed by summing $\mathbb{P}$ for all cards except the queen of hearts:

$$\mathbb{P}(\text{not } Q\clubsuit) = \mathbb{P}\Big(\underbrace{2\clubsuit \text{ OR } 3\clubsuit \text{ OR } 4\clubsuit \text{ OR } \ldots \text{ OR } 2\spadesuit \text{ OR } 3\spadesuit \text{ OR } 4\spadesuit \text{ OR } \ldots}_{\text{all cards except } Q\clubsuit} \Big)$$
$$= \mathbb{P}(2\clubsuit) + \mathbb{P}(3\clubsuit) + \mathbb{P}(4\clubsuit) + \cdots + \mathbb{P}(2\spadesuit) + \mathbb{P}(3\spadesuit) + \mathbb{P}(4\spadesuit) + \ldots$$

According to the indifference principle, each term on the right should be the same number x. Since there are 51 terms for the 51 non-$Q\clubsuit$ cards, we have

$$\mathbb{P}(\text{not } Q\clubsuit) = \underbrace{x + x + x + x + \cdots + x + x}_{51 \text{ terms}} = 51x$$

Furthermore, $\mathbb{P}(Q\clubsuit) = x$, since this card is just like any of the others. Thus, Eq. (10.7) becomes

$$x + 51x = 1 \tag{10.8}$$

which means $52x = 1$, so that $x = \boxed{\mathbb{P}(Q\clubsuit) = 1/52}$.

The above example illustrates an important idea:

> If there are N mutually exclusive possible outcomes, and none of the outcomes is favored over the others, than any one of these outcomes has a probability of $1/N$.

Put another way, the total probability of 1 is distributed evenly amongst the N mutually exclusive possibilities. We will use this idea quite a bit moving forward.

Example 10.5: Drawing Any Queen

Problem: What is the probability of drawing a queen from a shuffled deck of 52 playing cards?

Solution: This is really an *or* question, since we are asking for

$$\mathbb{P}\Big(Q\clubsuit \text{ OR } Q\spadesuit \text{ OR } Q\heartsuit \text{ OR } Q\diamondsuit\Big)$$

These are mutually exclusive events (e.g. when you draw a single card it can't be both a $Q\clubsuit$ and $Q\diamondsuit$), and hence the sum rule of Eq. (10.6) applies:

$$\begin{aligned}\mathbb{P}\Big(Q\clubsuit \text{ OR } Q\spadesuit \text{ OR } Q\heartsuit \text{ OR } Q\diamondsuit\Big) &= \mathbb{P}\Big(Q\clubsuit\Big) + \mathbb{P}\Big(Q\spadesuit\Big) + \mathbb{P}\Big(Q\heartsuit\Big) + \mathbb{P}\Big(Q\diamondsuit\Big) \\ &= \frac{1}{52} + \frac{1}{52} + \frac{1}{52} + \frac{1}{52} \\ &= \frac{4}{52}\end{aligned}$$

Thus, the probability of drawing a queen is $\boxed{\mathbb{P}(Q) = 4/52}$.

Note the same logic works for any card type, whether its a queen or a 5 or whatever.

The above example illustrates another important idea:

> Assume there are N mutually exclusive possible outcomes, none favored over the others. The probability of getting an outcome in some subset of n outcomes is n/N.

A few more examples of the sum rule:

Example 10.6: Drawing a 2 or a 7

Problem: What is the probability of drawing either a 2 or a 7 off the top of a shuffled deck of cards?

Solution: According to the results of Ex. 10.5, $\mathbb{P}(2) = \mathbb{P}(7) = 4/52$, since there are 4 suits of cards. Further note that you can't draw a card and simultaneously get a 2 and a 7, meaning these outcomes are mutually exclusive. We can then use Eq. (10.6) to obtain

$$\begin{aligned}\mathbb{P}\Big(2 \text{ OR } 7\Big) &= \mathbb{P}(2) + \mathbb{P}(7) \\ &= \frac{4}{52} + \frac{4}{52} \\ &= \frac{8}{52}\end{aligned}$$

and hence there is a probability of $\boxed{8/52 \approx 15\%}$ of drawing a 2 or a 7.

Example 10.7: The Case of the Missing Laptop

Problem: Your laptop is missing. You think there is a 20% chance that your neighbor Josh stole it, a 30% chance that your arch-enemy Mary stole it, a 10% chance that your roommate borrowed it without asking, and a 40% chance you just misplaced it and forgot where you put it. What is the probability that it got stolen?

Solution: The two theft possibilities are that Josh stole it, or Mary stole it. This is an *or* situation. However, we don't know ahead of time if Josh stealing it and Mary stealing it are mutually exclusive, because they could have stolen it together. The general sum rule Eq. (10.5) says

$$\mathbb{P}\Big(\text{Josh } \textsc{or} \text{ Mary}\Big) = \underbrace{\mathbb{P}\Big(\text{Josh}\Big)}_{0.20} + \underbrace{\mathbb{P}\Big(\text{Mary}\Big)}_{0.30} - \underbrace{\mathbb{P}\Big(\text{Josh } \textsc{and} \text{ Mary}\Big)}_{?}$$

Let's assume that Josh and Mary didn't conspire together to steal your laptop[a], so that $\mathbb{P}(\text{Josh } \textsc{and} \text{ Mary}) = 0$. The probability it was stolen is then $0.20 + 0.30 = \boxed{0.50 = 50\%}$.

[a] Though I wouldn't put it past them. I don't trust those two.

Example 10.8: Using Product and Sum Rule Together

Problem: What is the probability of drawing two aces off the top of a shuffled deck of playing cards? (The first ace is not put back in the deck before drawing the second).

Solution: We can rephrase the question as

> "What is the probability of drawing an ace off the top of a shuffled deck, **and** then drawing another ace off the top."

The presence of *and* here means we use the product rule. However, the probability of drawing any ace is an *or* question, since it could be the ace of spades or the ace of clubs etc.

Let A denote drawing the first ace and B denote drawing a second ace. There are 4 aces in a deck of 52 cards, so following the steps of Ex. 10.5, the sum rule gives

$$\mathbb{P}(A) = \mathbb{P}(\text{draw first ace}) = \frac{4}{52}$$

Once an ace is already drawn from the deck there are only 3 aces possible in a total of 51 remaining cards. Thus

$$\mathbb{P}\left(B|A\right) = \mathbb{P}\Big(\text{draw second ace} \,\Big|\, \text{already drew first ace}\Big) = \frac{3}{51}$$

Using Eq. (10.1), the product rule now gives

$$\underbrace{\mathbb{P}\left(\text{two aces}\right)}_{\text{first ace kept out of deck}} = \mathbb{P}\left(AB\right) = \underbrace{\mathbb{P}\left(A\right)}_{=\frac{4}{52}} \cdot \underbrace{\mathbb{P}\left(B|A\right)}_{=\frac{3}{51}} = \boxed{\frac{4\cdot 3}{52\cdot 51} \approx 0.0045}$$

Note that A happening influences the probability of B. Thus, A and B are *dependent* outcomes. One way of looking at this is to say that the deck "has a memory" of A happening, being that the deck is now short an ace. On the other hand, if we had put the first ace back into the deck and shuffled before drawing the second card, then A and B would be *independent*, and we would have $\mathbb{P}(A) = \mathbb{P}(B) = 4/52$.

10.3.3 Summing Over All Possibilities

If you add up the probabilities of all mutually exclusive possibilities, you obtain 1. This just means that we're certain that *something* happened, or that *some* explanation is correct. For example, in Ex. 10.7, if we assume the four possibilities stated are mutually exclusive, their probabilities add up to 1. This means that there are no other possibilities.

We can state this result as a compact equation. Instead of using different letters A, B, C, etc. to represent possible events/explanations, let's just use a single letter with subscripts, as we did in Sec. 1.4. The first possibility is denoted A_1, the second A_2, and so on, up to A_n. Then

$$\underbrace{\sum_{i=1}^{n} \mathbb{P}(A_i) = 1}_{\text{if the } A_i \text{ are mutually exclusive and cover all possibilities}} \tag{10.9}$$

Example 10.9: Competing Hypotheses

Problem: We are doing an experiment to test four mutually exclusive scientific hypotheses that we assume cover all possibilities. Denote them H_1, H_2, H_3, H_4. We conduct an experiment, and the data suggests that $\mathbb{P}(H_1) = 0.2$, $\mathbb{P}(H_2) = 0.35$, and $\mathbb{P}(H_3) = 0.1$. What is $\mathbb{P}(H_4)$?

Solution: Since H_1 through H_4 cover all possibilities and are mutually exclusive, their probabilites must add to 1 by Eq. (10.9). Thus

$$\sum_{i=1}^{4} \mathbb{P}(H_i) = \mathbb{P}(H_1) + \mathbb{P}(H_2) + \mathbb{P}(H_3) + \mathbb{P}(H_4) = 1$$

$$\longrightarrow \quad 0.2 + 0.35 + 0.1 + \mathbb{P}(H_4) = 1$$

so that $\mathbb{P}(H_4) = 1 - 0.2 - 0.35 - 0.1 = \boxed{0.35}$. We regard the most plausible hypotheses to be H_2 and H_4. These are each roughly 3.5 times as plausible as H_3, etc.

10.4 Exercises

1. The probability that you miss your alarm in the morning is 0.05. The probability that you forget your bookbag is 0.01 on a day when you hear your alarm (and hence wake up on time), but is 0.3 on a day when you miss your alarm and are rushed.

   i) What is the probability that you both miss your alarm *and* forget your bookbag?

   ii) What is the probability that you hear your alarm but still forget your bookbag?

   iii) What is the probability that your miss your alarm but still remember your bookbag?

2. What is the probability of drawing a card from a shuffled deck and getting either an ace, king, queen, or jack?

3. A weighted (unfair) six-sided die has the following probabilities for rolling a number: $\mathbb{P}(2) = \mathbb{P}(3) = \mathbb{P}(4) = 0.2, \mathbb{P}(5) = \mathbb{P}(6) = 0.05$. What is $\mathbb{P}(1)$?

4. What is the probability of rolling a number ≥ 5 when rolling a fair six-sided die?

5. A student is asked to compute the probability of rolling a six sided die and obtaining either a multiple of 3 or an even number. They figure that the probability of rolling a multiple of 3 is $\mathbb{P}(3 \text{ or } 6) = 2/6$, and that the probability of rolling an even is $\mathbb{P}(\text{even}) = 1/2$, since half the numbers on a die are even. They then add these two probabilities using the sum rule, and find that the probability of rolling a multiple of 3 or an even is $\mathbb{P}(\text{multiple of } 3) + \mathbb{P}(\text{even}) = 2/6 + 1/2 = 5/6$.

   The student has made an error in this computation. Find it and explain where they went wrong.

6. A fair coin is flipped 4 times in a row. Compute the probability of flipping

   i) 4 heads in a row

   ii) heads, heads, tails, and then heads, i.e. $\mathbb{P}(HHTH)$

   iii) heads, tails, heads, and then heads, i.e. $\mathbb{P}(HTHH)$ How does this compare to the previous answer?

   iv) 3 heads and 1 tail in any order.

   HINT: Think about all the possible sequences of 4 flips that would contain a total of 3 heads. Each sequence has a probability governed by the product rule, and if the sequences are mutually exclusive, the sum rule can give you an overall probability.

CHAPTER 11

Bayes' Theorem

Human beings always hold prior beliefs of some kind when encountering new circumstances. If we observe something that *contradicts* our belief, we may become a bit less certain of the belief. On the other hand, if we observe something that is *compatible* with our belief, we likely become more certain of our belief. This type of thinking process, which reflects in large part how we as humans think (and which is crucial in science), is captured mathematically in Bayes' Theorem.

11.1 Statement of Theorem

A simple yet far-reaching theorem in probability theory is Bayes' Theorem. The theorem is simply a restatement of

$$\mathbb{P}(A \text{ and } B) = \mathbb{P}(B \text{ and } A) \tag{11.1}$$

This statement is obvious: if A and B both happened (or are both true), then clearly it's the same thing as saying B and A both happened (or are both true). This obvious fact can be written in a less obvious way by invoking the product rule, Eq. (10.1), which gives that

$$\mathbb{P}(A \text{ and } B) = \mathbb{P}(B|A) \cdot \mathbb{P}(A)$$
$$\mathbb{P}(B \text{ and } A) = \mathbb{P}(A|B) \cdot \mathbb{P}(B)$$

so that we may rewrite Eq. (11.1) as

$$\mathbb{P}(B|A) \cdot \mathbb{P}(A) = \mathbb{P}(A|B) \cdot \mathbb{P}(B). \tag{11.2}$$

Dividing both sides of this equation by $\mathbb{P}(A)$ yields Bayes' Theorem:

$$\mathbb{P}(B|A) = \frac{\mathbb{P}(A|B)}{\mathbb{P}(A)} \cdot \mathbb{P}(B) \tag{11.3}$$

The real achievment of Thomas Bayes was not in writing down the obvious Eq. (11.1) and doing one line of algebra, but rather was in formulating the concept of *conditional probability* in Eq. (10.1).

where the various pieces in the formula are typically referred to by the following terms:

- $\mathbb{P}(B|A)$ is known as the `posterior probability` of B given A
- $\mathbb{P}(A|B)$ is known as the `likelihood` or probability of A given B
- $\mathbb{P}(A)$ is the `probability` of A
- the ratio of likelihood/probability $= \mathbb{P}(A|B)/\mathbb{P}(A)$ is called the `compatibility factor`
- $\mathbb{P}(B)$ is known as the `prior probability` of B

While we use the term *compatibility factor* in this book, note this term is not generally used in the field or in other books.

Let's now consider the meaning of these various pieces.

11.2 Meaning and Interpretation of Theorem

It's a bit easier to grasp the meaning of the formula if we consider the typical example of confronting a hypothesis H with data D. For this scenario, let's write Bayes' Theorem, Eq. (11.3), using the letters D and H instead of the generic letters A and B:

$$\mathbb{P}(H|D) = \frac{\mathbb{P}(D|H)}{\mathbb{P}(D)} \cdot \mathbb{P}(H) \tag{11.4}$$

posterior probability = plausibility of H *after* accounting for data D

compatibility factor = compatibility of data D with H

prior probability = plausibility of H *before* accounting for data D

Thinking of the formula from right-to-left, we start with what we consider to be the probability $\mathbb{P}(H)$ of our hypothesis H being true before (or prior) to accounting for information/data D. This prior probability gets multiplied by the compatibility factor $\mathbb{P}(D|H)/\mathbb{P}(D)$, which is a measure of how compatible the observed data D is with the hypothesis H. There are two possibilities for the compatibility factor:

Technically, a third possibility is $\mathbb{P}(D|H)/\mathbb{P}(D) = 1$. This would only come up if the data D doesn't really tell us anything about H – the hypothesis is just as probable in light of the new data D as it was before looking at D.

1. $\mathbb{P}(D|H)/\mathbb{P}(D) < 1$. This means our data D is *incompatible* to some extent with, or doesn't favor, our hypothesis H. Because we are multiplying our prior $\mathbb{P}(H)$ by a number smaller than 1, the resulting posterior probability $P(H|D)$ is *less* than the prior probability. Hence, in light of the new data D, the hypothesis H is now less plausible.

2. $\mathbb{P}(D|H)/\mathbb{P}(D) > 1$. This means our data D is *compatible* to some extent with, or favors, our hypothesis H. Because we are multiplying our prior $\mathbb{P}(H)$ by a number bigger than 1, the resulting posterior probability $P(H|D)$ is *more* than the prior probability. Hence, in light of the new data D, the hypothesis H is now more plausible.

We aren't, however, required to think of the A and B in Eq. (11.3) in terms of a hypothesis and data. A and B can be occurrences, events, explanations, facts, etc. In order to make these ideas more concrete, let's look at some examples.

Example 11.1: Using Bayes' Theorem

Problem: In recent years in the city of Bayesville, on average it has rained 73 days per year. Furthermore, your past experience (carefully recorded in your lab notebook) indicates that on days with no rain, there is nonetheless a 62% chance of it being cloudy.

You wake up one morning, and notice it is cloudy in Bayesville. What is the probability it will rain today?

Since we are told $\mathbb{P}(\text{clouds}|\text{no rain}) = 0.62$, we might be tempted to say *"this is the same thing as* $\mathbb{P}(\text{no rain}\,|\text{clouds}) = 0.62$, *meaning there is a 62% chance of no rain when its cloudy. This implies there is a 38% chance it does rain when its cloudy!"* as an answer to the problem. This is wrong, though, because generally speaking $\mathbb{P}(A|B) \neq \mathbb{P}(B|A)$. It's worth taking some time to think through why these aren't equal.

Solution: The generic letters A and B in Eq. (11.3) are easy to get mixed up, so let's use the letters $R =$ it rains, and $C =$ it's cloudy. We are interested in computing the probability that it rains given that we see clouds, i.e. $\mathbb{P}(R|C)$.

Bayes' Theorem then says

$$\mathbb{P}(R|C) = \frac{\mathbb{P}(C|R)}{\mathbb{P}(C)} \cdot \mathbb{P}(R) \tag{11.5}$$

Let's compute each of the pieces on the right-hand-side of Eq. (11.5).

- $\mathbb{P}(R)$ = the prior probability of rain, i.e. the probability it would rain on any given day, without further information. Since it has rained on average 73 days out of 365, it would be reasonable to assume a prior of $\mathbb{P}(R) = 73/365 = 0.20$, or 20%.
- $\mathbb{P}(C|R)$ = the likelihood (or probability) of clouds if one assumes that there is rain. Since there are basically *always* clouds when it rains, we'll just assume $\mathbb{P}(C|R) = 1 = 100\%$.
- $\mathbb{P}(C)$ = the probability that there are clouds. This denominator term is usually the most work to compute. Note that there could be clouds even if there is no rain. There could be clouds and rain, OR there could be clouds and no rain:

$$\begin{aligned}
\mathbb{P}(C) &= \mathbb{P}(\text{cloudy}) \\
&= \mathbb{P}\left(\begin{array}{c}\text{cloudy} \\ \text{AND} \\ \text{rain}\end{array} \quad \textbf{OR} \quad \begin{array}{c}\text{cloudy} \\ \text{AND} \\ \text{no rain}\end{array} \right) \\
&= \mathbb{P}(C \text{ AND } R) + \mathbb{P}(C \text{ AND not } R) \\
&= \mathbb{P}(C|R) \cdot \mathbb{P}(R) + \mathbb{P}(C|\text{not } R) \cdot \mathbb{P}(\text{not } R) \\
&= 1.00 \cdot 0.2 + 0.62 \cdot 0.8 \\
&= 0.696
\end{aligned}$$

  where we used $\mathbb{P}(\text{not } R) = 1 - \mathbb{P}(R) = 1 - 0.2 = 0.8$ according to the complement rule of Eq. (9.3).

Whenever he have an OR situation, we use the sum rule of Eq. (10.6), noting that rain and no rain are mutually exclusive.

To compute $\mathbb{P}(C \text{ AND } R)$ we use the product rule of Eq. (10.1).

Using these numbers, the compatibility factor is $\mathbb{P}(C|R)/\mathbb{P}(C) = 1/0.696 = 1.437$. This is greater than one, and hence the posterior probability of rain should be higher than the prior. This is common sense, being that clouds were observed.

Putting these values into Eq. (11.5), we find

$$\mathbb{P}(R|C) = \frac{1}{0.696} \cdot 0.2 = 0.287 = \boxed{28.7\%}$$

Before we woke up, our prior probability for rain would have been 20%. Having seen clouds, our posterior probability is 28.7%.

11.3 Extended Example 1: The Impact of Priors

An interesting and commonly given example of Bayes' Theorem involves diagnostic testing, and in some cases can yield surprising insights. Let's suppose that the horrible disease *statistitis* afflicts 0.3% of the population, and that a diagnostic

test for this tragic affliction has an accuracy of 98%. If you go to the doctor and test positive for statistitis, what can you conclude? Let's sort this out using Bayes' Theorem.

First of all, let's pose a precise question:

> **What is the probability I have the disease, given that I tested positive?**

Let's introduce some symbols:

- testing positive = "+", testing negative = "−"
- having the disease = "✓", not having the disease = "×"

Thus, in compact notation, we are interested in computing $\mathbb{P}(✓|+)$. Now let's use Bayes' Theorem, Eq. (11.3):

$$\mathbb{P}(✓|+) = \frac{\mathbb{P}(+|✓)}{\mathbb{P}(+)} \cdot \mathbb{P}(✓) \tag{11.6}$$

Let's look at the factors on the right-hand side of this equation:

✓ = has disease
× = doesn't have disease

+ = positive test
− = negative test

- The *prior probability* $\mathbb{P}(✓)$ is given in the problem as $0.3\% = 0.003$. This is known as the prior probability because it represents the probability, *prior* to any testing or further information, that a random person would have the disease.
- The *likelihood* of testing positive given that you actually have the disease, $\mathbb{P}(+|✓)$, is by definition the accuracy of the test, which is given as 98%=0.98.
- The *probability* of testing positive $\mathbb{P}(+)$ is the total probability of getting a positive result. There are two mutually exclusive ways to test positive: to have the disease and test positive, OR not have the disease and test positive. Using the product and sum rules, we have

$$\begin{aligned}\mathbb{P}(+) &= \mathbb{P}(+ \text{ AND } ✓) + \mathbb{P}(+ \text{ AND } ×) \\ &= \underbrace{\mathbb{P}(+|✓)}_{0.98} \cdot \underbrace{\mathbb{P}(✓)}_{0.003} + \underbrace{\mathbb{P}(+|×)}_{1-0.98} \cdot \underbrace{\mathbb{P}(×)}_{1-0.003} \\ &= 0.00294 + 0.01994 \\ &= 0.02288\end{aligned} \tag{11.7}$$

Note that the probability of not having the disease yet testing positive, $\mathbb{P}(+|×)$, i.e. of getting a false positive, is 2%, since the test is 98% accurate. We thus have a compatibility factor of

$$\frac{\mathbb{P}(+|✓)}{\mathbb{P}(+)} = \frac{0.98}{0.02288} = 42.83 \tag{11.8}$$

This compatibility factor is a good deal greater than 1, meaning that there is high compatibility between the data (that I tested positive) and the hypothesis (that I actually have the disease). Since we will be multiplying the prior by a number

much bigger than 1, we expect the posterior probability will be much greater than the prior probability that I have the disease.

Inserting our numbers into Eq. (11.6) yields

$$\mathbb{P}(\checkmark|+) = \frac{0.98}{0.02288} \cdot 0.003 = 42.83 \cdot 0.003 = 0.128 = \boxed{12.8\%}. \tag{11.9}$$

Thus, in light of the positive test, there is a 12.8% chance you actually have the disease. As expected, our posterior is much higher than the prior.

Wait - Only 12.8%?

Even though I tested positive, there is only a 12.8% chance I actually have the disease. It's far more likely that I'm actually statistitis-free! This may seem mysterious. In essence, there is only a 12.8% chance the test has correctly pegged me as having statistitis, even though the test is 98% accurate. What gives?

The issue here is the incredibly small prior probability $\mathbb{P}(\checkmark)$. Even though the test is pushing us in the direction of thinking I have the disease, beforehand the probability of this was considered to be very small. Thus, it's far more likely that my test result was a false positive than a true positive. For example, imagine that 1,000 people are randomly selected from the population. On average, only 3 of these people will actually have the disease, if the prior probability is to be believed. Thus there will be 997 people without the disease. Let's look at all the positive test results in our group of 1000 people:

- There is a 98% chance that a truly infected person will test positive. There are on average 3 truly infected people per 1000. Thus, on average, there will be $3 \cdot 0.98 = 2.94$ true-positives in a group of 1,000 random people.
- There is a $100 - 98 = 2\%$ chance that a non-infected person will test positive (the test has failed). Since there are 997 non-infected people, there will be on average $0.02 \cdot 997 = 19.9$ false-positives in a group of 1000 random people.

Thus, we can see that there are roughly 19.9/2.94=6.7 times as many false positives as there are true positives. This is reflected in Eq. (11.7), where the 2nd term is 6.7 times bigger than the first. Hence, if I test positive, without knowing anything else, I'm more likely to be a false-positive than a true-positive.

If you inspect Eq. (11.7) closely, you'll see it's really doing the same math we did in analyzing the group of 1,000 random people.

Some Lessons and Conclusions

A general conclusion to be drawn here is that a *single* diagnostic test is largely worthless if its false-positive rate (here 2%) is considerably larger than the true percentage of the affected population (here 0.3%). An even more general conclusion is that *a little bit of data won't convince us of anything if our prior belief in that thing is very small.* Whenever the prior probability is small, we need lots and lots of evidence to conclude something else. This pretty closely models a lot of how we think in everyday life. If we already believe something, we don't need much (if any) evidence. However, if we strongly disbelieve something, it's going to take more than just a little bit of evidence to sway us.

Multiple tests *can* tell you something, though. After all, if you're actually disease free, the chance of the test being wrong twice is presumably 0.02×0.02, unless there's something about your body chemistry that makes the test more likely to be wrong, etc.

A final point: where did the original $\mathbb{P}(\checkmark)$ number of 0.3% come from? Maybe it was just a guess. Maybe it came from doing the very same test we're criticizing. Maybe it came from a much more accurate (and expensive) version of the

test. Maybe something else. Such prior probabilities can have a variety of different sources, and some of them are more believable than others. In some cases it is literally just a guess, which is one of the valid critiques made of Bayesian methods. There are various ways of mitigating this problem, such as collecting so much data that the prior probability doesn't affect the final conclusion very much. However, this isn't always possible, and at any rate there is often no way around the general principle that *if you feed garbage assumptions or information into a theoretical/-computational model, you may get garbage conclusions coming out the other end.*

While it is a bit beyond the scope of this book, a proper Bayesian analysis would examine how sensitive our results are to our initial guesses or choice of prior. If changing the prior just a tiny bit results in a very different posterior, we may be skeptical of our original results, since being just a tiny bit wrong about the prior would change the results a lot. On the other hand, if changing the prior a bit doesn't change the final answer too much, we have more confidence our posterior probability is correct even if we're slightly wrong about the prior. A full analysis would assign numbers to things like "a bit" and "a little" and "too much".

11.4 Extended Example 2: Bayesian Updating

This panda example is given in Richard McElreath's text *Statistical Rethinking* [1].

Imagine there are two species of pandas, A and B, that look identical and that have generally equal numbers in the wild. There is one difference between them: a female in species A is 10% likely to give birth to twins, while a female in species B is 20% likely to give birth to twins.

If we witness a female panda give birth to twins, what is the probability they are species A? In other words, what is $\mathbb{P}(A|\text{twins})$? Bayes' Theorem gives

$$\mathbb{P}(A|\text{twins}) = \frac{\mathbb{P}(\text{twins}|A) \cdot \mathbb{P}(A)}{\mathbb{P}(\text{twins})} \tag{11.10}$$

Let's compute each of the factors on the right-hand side:

- The *prior probability* of being species A is 0.5, since we are told A and B are equal in numbers.
- The *likelihood* $\mathbb{P}(\text{twins}|A)$ of giving birth to twins for species A is given as 10%=0.1.
- The probability of twins $\mathbb{P}(\text{twins})$ is the total probability that a birth would be twins. This can happen in two mutually exclusive ways: the panda is species A and gives birth to twins, OR the panda is species B and gives birth to twins. Thus, we compute

$$\begin{aligned}\mathbb{P}(\text{twins}) &= \mathbb{P}(\text{twins AND } A) + \mathbb{P}(\text{twins AND } B) \\ &= \underbrace{\mathbb{P}(\text{twins}|A)}_{0.1} \cdot \underbrace{\mathbb{P}(A)}_{0.5} + \underbrace{\mathbb{P}(\text{twins}|B)}_{0.2} \cdot \underbrace{\mathbb{P}(B)}_{0.5} \\ &= 0.15\end{aligned}$$

Inserting these numbers into Eq. (11.10) gives

$$\mathbb{P}(A|\text{twins}) = \frac{0.1 \cdot 0.5}{0.15} = 0.\bar{3} \approx 33\% \tag{11.11}$$

It makes perfect sense that the panda is twice as likely to be species B. Since species B is twice as likely to have twins, and we observe twins, the panda being species B is twice as plausible as the panda being species A.

In other words, there is a posterior probability (i.e. probability after the data is taken into account) of 1/3 that the panda is species A and hence a 2/3 probability it is species B. The panda is thus twice as likely to be from species B as it is from species A.

Now Let's Do Bayesian Updating

For a future computation, *the new prior probability* $\mathbb{P}(A)$ *would be our old posterior probability* $1/3$. For example, consider the following scenario: you observed the panda having twins, you computed $\mathbb{P}(A|\text{twins}) = 1/3$. A few year later, this very same panda gives birth to a single cub. Given this new information, what is the probability that the panda is species A?

Well, we wish to compute $\mathbb{P}(A|\text{single})$. Bayes' Theorem gives

$$\mathbb{P}(A|\text{single}) = \frac{\mathbb{P}(\text{single}|A) \cdot \mathbb{P}(A)}{\mathbb{P}(\text{single})} \tag{11.12}$$

and we again look at the factors on the right-hand side:

- the prior probability $\mathbb{P}(A)$, i.e. the probability our panda is species A prior to looking at the *new* information, is $1/3$, the result of our previous calculation.
- the likelihood $\mathbb{P}(\text{single}|A)$ is $1 - \mathbb{P}(\text{twins}|A) = 0.9$, assuming triplets and above are super-unlikely.
- the probability $\mathbb{P}(\text{single})$ is the total probability of having a single cub. This can happen in two mutually exclusive ways: species A has a single cub, or species B has a single cub. Thus,

$$\begin{aligned}\mathbb{P}(\text{single}) &= \mathbb{P}(\text{single and } A) + \mathbb{P}(\text{single and } B) \\ &= \underbrace{\mathbb{P}(\text{single}|A)}_{0.9} \cdot \underbrace{\mathbb{P}(A)}_{1/3} + \underbrace{\mathbb{P}(\text{single}|B)}_{0.8} \cdot \underbrace{\mathbb{P}(B)}_{2/3} \\ &= \frac{5}{6} = 0.8\bar{3}\end{aligned} \tag{11.13}$$

  where $\mathbb{P}(\text{single}|B)$ is one minus the probability of B having twins.

Inserting this all into Eq. (11.12) gives

$$\mathbb{P}(A|\text{single}) = \frac{0.9 \cdot \frac{1}{3}}{5/6} = 0.36 \tag{11.14}$$

We now find it 36% plausible that this panda is species A. Note that each time we acquire data, we *update* or revise our probability. At the very beginning, before we had any data, we said $\mathbb{P}(A) = 0.5$. After observing twins, which is more likely to happen with species B, we lowered our probability to $\mathbb{P}(A) = 0.\bar{3}$. After observing her giving birth to a single cub, which is slightly more likely to happen with species A, we slightly raised our probability to $\mathbb{P}(A) = 0.36$. It is helpful to organize the results of such a computation via a table such as Table 11.1.

data	favors	$\mathbb{P}(A)$	$\mathbb{P}(B)$
–	–	0.5	0.5
twins	B	$\downarrow$ 0.33	$\uparrow$ 0.67
single	A	$\uparrow$ 0.36	$\downarrow$ 0.64

Table 11.1: Bayesian updating in the Panda example. The first data (twins) favors B, leading to updated probabilities in which $\mathbb{P}(B)$ increases while $\mathbb{P}(A)$ decreases. The next piece of data is that a single cub is born. This slightly favors A, leading to updated probabilities in which $\mathbb{P}(A)$ goes up slightly, while $\mathbb{P}(B)$ goes down slightly.

Bayesian updating of probabilities is a really nice way of incorporating information from previous experiments/studies into new ones.

11.5 Exercises

1. On any given day there is a 0.5% chance of a dangerous fire on your property. Assume there is a 10% probability of seeing smoke on your property on any given day (since there are a lot of neighbors who barbecue). Further assume that 90% of dangerous fires make smoke. If you see smoke near your house, what is the probability there is a dangerous fire on your property?

2. Redo the computation of $\mathbb{P}(\checkmark|+)$ of Sec. 11.3 assuming that 10% of the population has the disease. Interpret your result, and compare with the situation given in the test (where 0.3% of the population has the disease). In which case is the test more adequate?

3. Reconsider the set-up of Sec. 11.4. Imagine you work at a zoo, and a new panda arrives.

   i) You see the new panda give birth to a single cub. What is the posterior probability this panda is of species B?

   ii) A few years later, you see the same panda give birth to another single cub. Again, what is the updated posterior probability this panda is of species B?

   iii) Finally, after giving birth to the two single cubs, you see the same panda give birth to twins. What is the updated posterior probability this panda is of species B?

4. There are 3 wocket-making machines (call them A, B, and C) in a wocket factory. You know that 20% of the wockets produced by the factory are made in machine A, 30% are produced by machine B, and the rest are produced by machine C. However, these machines aren't perfect, and they sometimes produce defective parts. Measurements show that 6% of the wockets made by machine A are defective, while 2% of the machine B wockets are defective, and only 1% of the machine C wockets are defective.

   If a defective wocket is randomly selected from the total output of the factory, what is the probability this wocket was produced by machine A? What is the probability it was produced by machine B?

CHAPTER 12

Binomial Probabilities

As illustrated in Fig. 8.10, noise comes from a bunch of random external factors (that we don't measure) pushing the measurement one direction or another. In other words, noise is built up of a large number of random "pushes" all happening together. Thus, to deal with the statistical uncertainty due to noise in the data, we need to compute the probability of a bunch of different things happening together. This will ultimately lead us to the idea of a *probability distribution*, discussed in subsequent chapters. As a first step in all of this, in this chapter we will discuss computing the probability of a bunch of binomial events.

12.1 Computing A Binomial Probability

A `binomial event` is one in which the outcome has only two possibilities. A prototypical example is flipping a coin, where the result is either heads or tails. Another example would be to do testing on a sample of water to decide if bacteria counts are above a certain level, where the result is "yes" or "no".

Imagine we have an unfair coin where the probability of flipping heads is $\mathbb{P}(H) = 0.7$. We want to know the probability of flipping the coin 5 times and obtaining a total of 3 heads (in any order). To begin answering the question, let's look at one particular way to flip 3 heads: the sequence HTHTH. Each flip is independent of the previous flip, so we can use Eq. (10.4):

$$\begin{aligned}
\mathbb{P}(HTHTH) &\equiv \mathbb{P}\left(\begin{array}{ccccccccc} 1^{\text{st}}\text{ flip} & & 2^{\text{nd}}\text{ flip} & & 3^{\text{rd}}\text{ flip} & & 4^{\text{th}}\text{ flip} & & 5^{\text{th}}\text{ flip} \\ = H & \text{AND} & = T & \text{AND} & = H & \text{AND} & = T & \text{AND} & = H \end{array}\right) \\
&= \mathbb{P}(H) \cdot \mathbb{P}(T) \cdot \mathbb{P}(H) \cdot \mathbb{P}(T) \cdot \mathbb{P}(H) \\
&= 0.7 \cdot 0.3 \cdot 0.7 \cdot 0.3 \cdot 0.7 \\
&= 0.7^3 \cdot 0.3^2 \\
&= 0.03087
\end{aligned}$$

There is something important to note here: because you can multiply numbers in any order, *any* sequence of 5 flips with a total of 3 heads will have this same probability. To find the total probability of getting 3 heads (in any order), we simply add the probabilities of each compatible sequence in accord with Eq. (10.6):

$$\mathbb{P}\left(3 \text{ heads}\right) = \underbrace{\mathbb{P}(HTHTH)}_{0.7^3 \cdot 0.3^2} + \underbrace{\mathbb{P}(HTTHH)}_{0.7^3 \cdot 0.3^2} + \underbrace{\mathbb{P}(HHHTT)}_{0.7^3 \cdot 0.3^2} + \cdots$$

With some effort you'll find there are 10 distinct sequences with 3 heads. The

These sequences are HHHTT, HHTHT, HHTTH, HTHHT, HTHTH, HTTHH, THHHT, THHTH, THTHH, TTHHH.

probability we seek is thus

$$\begin{aligned}\mathbb{P}(3 \text{ heads}) &= (\#\text{ of sequences with 3 H}) \cdot 0.7^3 \cdot 0.3^2 \\ &= 10 \cdot 0.7^3 \cdot 0.3^2 \\ &= 0.3087\end{aligned}$$

In fact, we can immediately generalize the above computation. Notice that $0.7^3 = p^k$, where p is the probability of heads and k is the desired number of heads, and also that $0.3^2 = (1-p)^{n-k}$, where n is the total number of flips. Thus, when flipping a coin n times, the probability of getting k heads (in any order) is

$$\mathbb{P}(k \text{ heads}) = (\#\text{ of sequences with } k \text{ H}) \cdot p^k \cdot (1-p)^{n-k} \tag{12.1}$$

Unfortunately, as the # of flips increases, it becomes increasingly difficult to count the # of sequences manually. Fortunately, there is a well-known solution.

12.2 Efficient Counting

In a sequence of n items, where k of the items are one-thing (here heads) and the other $n-k$ items are another thing (here tails), the number of distinct sequences is given by the formula

For a derivation of the well-known Eq. (12.2), see for example the Math Is Fun website [8].

$$\#\text{ of sequences} = \frac{n!}{k!\,(n-k)!} \tag{12.2}$$

where an exclamation mark "!" means *factorial*, and is defined as

$$n! \equiv n \cdot (n-1) \cdot (n-2) \cdot (n-3) \cdots 2 \cdot 1 \tag{12.3}$$

For example, $5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 120$. Thus, referring back to Eq. (12.1), we have that

$$\mathbb{P}(k \text{ heads}) = \frac{n!}{k!(n-k)!} \cdot p^k \cdot (1-p)^{n-k} \tag{12.4}$$

In the particular case of $n = 5, k = 3$ from earlier, the number of sequences is given by

$$\frac{n!}{k!(n-k)!} = \frac{5!}{3!2!} = \frac{5 \cdot 4 \cdot 3 \cdot 2 \cdot 1}{3 \cdot 2 \cdot 1 \cdot 2 \cdot 1} = \frac{120}{12} = 10$$

which is the same result we found by just listing out all the sequences.

We can compute factorials in R via the command `factorial()`. Thus, to compute 10! we could issue `factorial(10)` and obtain 3,626,800.

Example 12.1: Counting Sequences

Problem: Waniya samples 40 rainbow trout from a local stream, and it turns out that a total of 27 of them weigh more than 20 pounds. In her notebook she records M for "more than 20 lbs" and L as "less than 20 lbs". Thus, she records a total of 27 M and 13 L. How many different 40-length sequences are compatible with her totals?

Solution: Instead of heads and tails, it's M and L, but this is still binary data. Thus, we can use Eq. (12.4), except now k refers to the number of M. We have

- $n =$ total # of fish $= 40$
- $k =$ # of fish over 20 lbs $= 27$

and hence the number of compatible sequences is

$$\begin{aligned} \#\text{ seq} &= \frac{n!}{k!(n-k)!} \\ &= \frac{40!}{27!(40-27)!} \\ &= \texttt{factorial(40)/(factorial(27)*factorial(40-27))} \\ &= 12,033,222,880 \end{aligned}$$

or over 12 billion sequences. Using R here saves a tiny bit of time compared to trying to write them all out and count by hand.

Note the parentheses in the denominator of the R expression are required. If we want $\frac{a}{b \cdot c}$, we need to type `a/(b*c)`. If we instead type `a/b*c`, we are actually computing $\frac{a}{b} \cdot c$ according to order of operations.

12.3 Binomial Experiments

We've been talking a lot about coin-flips, but the ideas discussed here are much broader. The math applies to any sequence of independent binomial events. Here's one possible example: suppose it is known that 2% of a specific species of trees in a forest is infected with some disease. If I select a random sample of 50 trees, what is the probability that 7 of the trees are infected?

Instead of flipping a coin, we're testing a tree. Instead of getting heads or tails with probabilities 0.7 or 0.3, we'll get positive or negative for the disease with probabilities 0.02 and 0.98. The mathematical description is actually the same as that of equation Eq. (12.4). Thus,

$$\mathbb{P}\left(k \text{ trees infected}\right) = \frac{n!}{k!(n-k)!} p^k (1-p)^{n-k} \tag{12.5}$$

and in our case we have the number of trees $n = 50$, the probability of being infected as $p = 0.02$, and we're looking at the specific case of $k = 7$ infected trees. Putting in the numbers we have

$$\begin{aligned} \mathbb{P}\left(k \text{ trees infected}\right) &= \frac{50!}{7! \cdot 43!} 0.02^7 \cdot 0.98^{50-7} \\ &= \frac{50 \cdot 49 \cdots 45 \cdot 44}{7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1} 0.02^7 \cdot 0.98^{43} \\ &= 5.36 \cdot 10^{-5} \\ &= 0.0000536 \\ &= 0.00536\% \end{aligned} \tag{12.6}$$

where the second line follows from canceling all the like factors in the numerator and denominator. We can also compute this number using R:

(R.12.1)
```
n <- 50
k <- 7
p <- 0.02
factorial(n)/(factorial(k)*factorial(n-k))*p^k*(1-p)^(n-k)
```

Is Our Formula Valid?
It is important to note that Eq. (12.4) is only valid if the binomial events are *independent*. For coin flips this is pretty sensible, as a coin has no way of "remembering" what happened during the previous flip, and hence the outcome of a flip shouldn't depend on a previous flip. It is NOT as clear that a tree being infected is independent of whether a neighboring tree is infected. For some types of disease transmission it would be absurd and totally incorrect to assume independence in this way. Thus, in many cases it may be completely incorrect to model our situation with independent binomial probabilities. For the sake of the current example we're assuming we selected the trees in a random enough fashion that it's okay to assume independence. But our assumption might be radically incorrect, and in general one must be mindful of the restrictions and assumptions of a formula or method.

If the disease rate is really 2%, we see that it's extremely unlikely to observe 7 out of 50 trees being diseased. If we had actually done the experiment and measured 7 sick trees, it might make us suspicious of the 2% number. Of course it's *possible* that even if the disease rate is actually 2% we could still find this many sick trees. So we haven't "disproved" the 2% number or anything, though we may want to recheck it. Another possibility is that the 2% number is correct but that our formula is invalid for the problem at hand (see margin-note).

A Logical Error We Made

The logic of our analysis above was that Eq. (12.6) is a small number, and hence we might doubt the disease rate is really 2%. This reasoning is incomplete. If the disease rate were really 2%, then we would expect only 1 of 50 trees on average to be diseased. But if we compute the probability of this, we find it is only 37%. In other words, even the *most* likely thing doesn't happen most of the time, simply because the probability is spread out amongst 51 different values of k. **We weren't being careful about what we were comparing $\mathbb{P}(k = 7)$ to.**

In this case the probability that k would be *7 or more* is still ridiculously small compared to, say, $\mathbb{P}(1) = 37\%$. For this reason, it is correct to be doubtful of the 2% number.

Example 12.2: Binomial Elephants

Problem: Suppose the elephant population in a certain geographic region is 57% female. If you encounter a group of 8 elephants, what is the probability that 6 are female and 2 are male?

Solution: Assuming that biological elephant sex is binary, M or F, we could try to adapt Eq. (12.4) by thinking of heads H as female "F" and tails T as male "M". We would have

- $p = \mathbb{P}(F) = 0.57$
- $q = \mathbb{P}(M) = (1 - p) = 0.43$
- $n = \#$ of elephants $= 8$
- $k = \#$ of $F = 6$

Plugging this in to Eq. (12.4), we get

$$\mathbb{P}(6F, 2M) = \frac{8!}{6!(8-6)!} \cdot 0.57^6 \cdot 0.43^2 = 0.178$$

Unfortunately, just because we plug numbers into some equation doesn't mean that what we're doing makes sense. Note that Eq. (12.4) assumes that the binary events are *independent* of each other. However, since elephants are matriarchal herd animals, any *group* is overwhelmingly likely to be mostly female (there may be younger males present).

Hence, modeling this situation as independent binomial events is almost certainly incorrect, as this method of obtaining a sample (just find a group) isn't sufficiently randomizing. Observing only *solitary* elephants would have the same issue, as these would be overwhelmingly male.

Our calculation would be correct only if it pertained to truly random sampling, which can be difficult (if not impossible) to attain in some situations. In fact, the very notion of *random sampling* isn't always well-defined. We'll return to this problem later in the book.

12.4 Exercises

1. An unfair coin has a probability $\mathbb{P}(H) = 0.55$ of being flipped heads.

   i) If you flip the coin 100 times, what is the probability of obtaining 55 heads?

   ii) A student makes the following claim: *"The probability of flipping heads is 55%. This means that if I flip the coin 100 times, I should get 55 heads. Thus, there is a 100% probability of getting 55 heads in 100 flips."*
   Analyze their reasoning, indicating in detail why you think it is correct or incorrect.

2. This problem involves testing for bacterial counts in water sources, as is done in the Water Quality Lab at SBC.

   Background Information

   A typical test for the bacteria count in a water sample involves a tray with 48 wells (as shown in Fig. 12.1). A 100 mL water sample is treated with chemicals that will fluoresce (glow) in the presence of bacteria, and the water is then poured into all the wells in the tray. The tray is incubated, and 24 hours later the tray may be examined under a blacklight. If the amount of bacteria in a specific well is above a certain threshold, the well will fluoresce. What follows is a highly simplified account of how such test results are analyzed.

   Even if a water sample comes from safe-to-drink water, there is still a *chance* that an unsafe amount of bacteria will happen to be clumped together in the sample and/or subsamples (i.e. single wells). *Assume that the probability of a well fluorescing for a sample of just-barely-safe-to-drink water is 20%.*

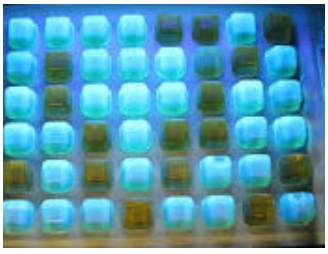
   

   Figure 12.1: A testing tray for water bacterial count, of the kind used in the Sitting Bull College water quality testing laboratory. Note that 32 of the 48 wells are fluorescing (glowing) under exposure to blacklight, indicating that these wells contained an unsafe amount of bacteria prior to incubation.

   Problem

   Imagine that somebody provides you with a sample of water, and you perform the tray test.

   i) What is the probability that a just-barely-safe-to-drink sample would result in having 32 of the 48 wells fluoresce, as shown in Fig. 12.1?

   ii) What would you conclude about the safety of the water for drinking?

   iii) Now imagine instead that only 9 of the 48 cells fluoresced. What is the probability of this happening for just-barely-safe-to-drink water? What would you conclude about its safety?

3. Reconsider the scenario in Sec. 12.3, where the disease rate for a tree is 2%. You intend to examine a random sample of 80 trees in a region of interest.

i) What is the probability that you find 2 infected trees?

ii) What is the probability that you find 10 infected trees?

iii) Write an R script that first defines `k` be a vector containing the integers 0 through 80. These integers represent the different possibilities for the number of infected trees you may find. Next compute the vector `P` which gives the binomial probability for each value of `k`. Make a plot of `P` vs `k`.

In other words, the first element of `P` should contain the probability of finding 0 infected trees, the second element of `P` should contain the probability of finding 1 infected tree, the third element of `P` should contain the probability of finding 2 infected trees, and so on.

CHAPTER 13

Discrete Probability Distributions

Repeating previous comments: as illustrated in Fig. 8.10, noise comes from a bunch of random external factors (that we don't measure) pushing the measurement one direction or another. Thus, to deal with the statistical uncertainty due to noise in the data, we need to compute the probability of a bunch of different things happening together. We do this using *probability distributions*.

13.1 The Binomial Distribution

The probability of obtaining k successes in n trials of a binomial experiment with success probability p is given by Eq. (12.4) or Eq. (12.5) as

$$\mathbb{P}(k) = \frac{n!}{k! \cdot (n-k)!} \cdot p^k \cdot (1-p)^{n-k} \tag{13.1}$$

Remember that *binomial experiment* means some type of experiment in which you perform a sequence of n trials where each trial outcome has two possible values each with fixed probability.

Consider again the example of flipping a weighted coin, with $\mathbb{P}(\text{heads}) = 0.7$ and $\mathbb{P}(\text{tails}) = 1 - \mathbb{P}(\text{heads}) = 0.3$. If we flip such a coin $n = 10$ times, the probability of flipping k heads is given by

$$\mathbb{P}(k) = \frac{10!}{k!\,(10-k)!} \cdot 0.7^k \cdot 0.3^{10-k} \tag{13.2}$$

You should verify that the probability of flipping 3 heads is $\mathbb{P}(3) \approx 0.009$. Recalling Sec. 12.3, we may compute this number in R through the commands

(R.13.1)
```
1 n <- 10
2 k <- 3
3 p <- 0.7
4 factorial(n)/(factorial(k)*factorial(n-k))*p^k*(1-p)^(n-k)
```

Because most functions in R are vectorized, the last line will be properly interpreted even if k is a vector rather than a single number. Hence, we could change line 2 of (R.13.1) to

(R.13.2)
```
2 k <- seq(0,10,1)
```

in order to calculate the probability for all values of k simultaneously, with results shown in Table 13.1.

# heads	probability (%)
0	0.00059
1	0.01379
2	0.14467
3	0.90017
4	3.67569
5	10.29193
6	20.01209
7	26.68279
8	23.34744
9	12.10608
10	2.82475

Table 13.1: Binomial probabilities for $n = 10, p = 0.7$.

Note that the total probability of 100% is *distributed*, or spread out amongst, the various values of k. Eq. (13.1) allows us to compute how much probability is given to each value of k and is known as a `probability distribution`. Eq. (13.1) in

particular is known as a *discrete* distribution because k is discrete, e.g. you can't flip 5.4 heads. In general, a probability distribution is an equation that indicates how the total probability of 100% is spread out amongst all the possible outcomes.

13.2 The `dbinom()` Function in R

Eq. (13.1) is used so much that it is built into R as the `dbinom()` function. Thus, in place of line 4 in (R.13.1), we could instead type

```
4  dbinom(x=k,size=n,prob=p)
```
(R.13.3)

We could omit typing the function parameters and instead just issue the command `dbinom(k,n,p)`. This is fine to do in the console, but a script is usually easier to read and error-check if you include the parameter names.

The numbers are the same ones we obtained in Table 13.1.

Example 13.1: Using `dbinom()`

Problem: 53% of a local bird population are female. What is the probability that a random sample of 40 birds will contain 25 females?

Solution: We will make two approximations / assumptions:

1. biological sex is binary, M or F
2. the sex of the birds are *independent* of each other

If we relax the approximation that biological sex is binary (say we expand the categories to include intersex), there are now more than 2 possible outcomes to a measurement (i.e. M, F, and I), and we would have to use the *multinomial distribution*, which we won't cover in this book.

With these two assumptions, we may use Eq. (13.1). There are a total of $n = 40$ birds, with probability $p = 0.53$ of being female. We are interested in the specific possibility that $k = 25$ of them are female. The R code

```
n <- 40          #number of observed birds
k <- 25          #number of females
p <- 0.53        #probability of being female
dbinom(x=k,size=n,prob=p)
```

We could combine these 4 lines into the shorter command `dbinom(25,40,0.53)`, but in a script more detail is generally preferred.

gives $\boxed{\mathbb{P}(k = 25) = 0.062}$.

Example 13.2: Plotting A Binomial Distribution

Problem: Make a plot of the binomial distribution (R.13.3), where $n = 10$ and $p = 0.7$.

Solution: The R code

```
n <- 10
k <- seq(0,10,1)
p <- 0.7
prob.vals <- dbinom(x=k,size=n,prob=p)
plot(x=k, y=prob.vals,
     main="Binomial Probabilities",
     xlab="number of heads", ylab="probability",
     panel.first=grid())
```

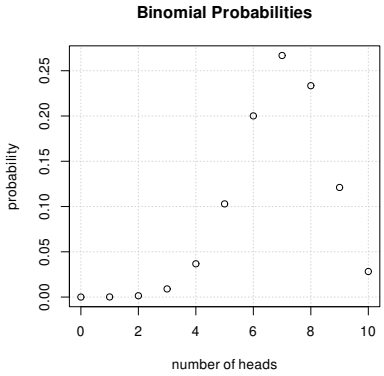


Figure 13.1: Plot of binomial probabilities for $n = 10, p = 0.7$ using `dbinom()`.

produces the plot shown in Fig. 13.1. This plot gives a quick visual for how the total probability of 1 is spread around the various possible number of heads k. The highest probabilitiy belongs to $k = 7$ heads in 10 flips, consistent with the heads probability of $p = 0.7$. By issuing the command `sum(prob.vals)` we can verify that $\sum_{k=0}^{n} \mathbb{P}(k) = 1$.

13.3 Any Single Outcome Is Unlikely

In Sec. 12.3 we mentioned that it is erroneous to assume that any specific outcome is all that likely. Let's further consider the example of $n = 10, p = 0.7$. Referring to Fig. 13.1 and Tab. 13.1, we see that even though $p = 0.7$, there is only a 27% probability of getting exactly 7 heads. Note that $\mathbb{P}(6)$ and $\mathbb{P}(7)$ are about the same size as $\mathbb{P}(7)$.

Thus, if you are told it is a $p = 0.7$ coin, and you flip 6 heads, it would be *erroneous* to claim

> Well, there is only a 20% chance of getting 6 heads in 10 flips for a $p(H) = 0.7$ coin, so we should be suspicious of the $p = 0.7$ number.

This doesn't make sense, because what are you comparing the 20% against? After all, there is only a 27% chance we get 7 heads. When there are a large number of possible outcomes, it is typically the case that if we only focus on a single specific outcome, it will always be at least somewhat improbable.

Thus, it often makes sense to consider groups of possible outcomes. For example, what if somebody flipped 5 or less heads? Would this be strong evidence against the idea that $p = 0.7$? You can verify that this probability is $\mathbb{P}(0) + \mathbb{P}(1) + \cdots + \mathbb{P}(5) = 0.15$. This isn't really strong evidence against the idea that $p = 0.7$. When the number of flips is small, flipping 50% or less heads (instead of the 70% we might expect) isn't all that unlikely. What about a larger number of flips? Let's look at this in an example problem.

Example 13.3: Probability of k or Less Heads

Problem: A weighted coin has a probability of flipping heads $p = 0.7$. The coin is flipped 1000 times. What is the probability of getting 500 or less heads?

Solution: We first obtain a vector of probabilities:

```
n <- 1000
k <- seq(from=0,to=1000,by=1)
p <- 0.7
prob.vals <- dbinom(x=k,size=n,prob=p)
```

The probability of flipping 500 or less heads, meaning the probability of flipping 0 heads OR 1 head OR 2 heads OR 3 heads etc. is

$$\mathbb{P}(k \leq 500) = \mathbb{P}(0) + \mathbb{P}(1) + \mathbb{P}(2) + \cdots + \mathbb{P}(499) + \mathbb{P}(500)$$

But these 501 probabilities are simply the first 501 elements of the vector `prob.vals`. We can get a vector of just those 501 values using the square bracket notation of Sec. 5.1, i.e. `prob.vals[1:501]`. We can then add all of these values using the `sum()` function. The probability we want is

```
5  sum(prob.vals[1:501])
```

which is the extraordinarily tiny number $\boxed{6.1 \times 10^{-40}}$. The probability of flipping 500 or less heads is effectively zero, and if it actually happened it would constitute *overwhelming* evidence that either the data is somehow suspect, or that p cannot actually be 0.7.

13.4 Cumulative Probability and `pbinom()`

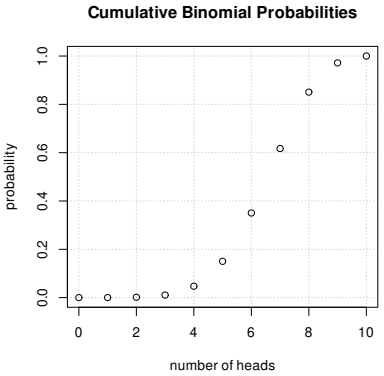


Figure 13.2: Plot of cumulative binomial probabilities for $n = 10, p = 0.7$ using `pbinom()`.

The `cumulative probability distribution` gives the probability of flipping k or less heads. In Ex. 13.3, we computed $\mathbb{P}(k \leq 500)$, which is the cumulative probability for $k = 500$. Instead of issuing lines 4 and 5 from Ex. 13.3, we could use the built-in R function for the cumulative binomial probability, `pbinom()`. The code `pbinom(q=k,size=n,prob=p)` would also return 6.1×10^{-40}.

We can make plot of the cumulative probability distribution. The case of $n = 10, p = 0.7$ is shown in Fig. 13.2. The commands to generate this plot are identical to those from Fig. 13.1 with the replacement `dbinom()` → `pbinom()`. Does it make sense to you that the probability approaches one as the number of heads approaches 10?

Example 13.4: Using `pbinom()`

Problem: A biologist studying adult male bison classifies them into two categories: small (less than 1500 lbs), and large (greater than 1500 lbs). Assume that, on average, 42% of bison are small. If the biologist studies a random sample of 30 adult bison, what is the probability of finding 20 or more large bison?

Solution: Since we are classifying weight as either S (small) or L (large), this is binomial data. If we further assume the weight of the bison are independent of each other, we may assume the situation is described by Eq. (13.1). However, note that cumulative probability is defined to be *less than or equal* to a number, whereas we are asked a question *greater than or equal* to a number. Hence, we will reformulate the question by recognizing that for our sample of 30 bison,

"20 or more large bison" = "10 or less small bison"

Thus, we'll compute the probability of finding <= 10 small bison. The code

```
n <- 30          #number of observed bison
k <- 10          #max number of small
p <- 0.42        #probability of being small
pbinom(q=k,size=n,prob=p)
```

gives $\mathbb{P}(S \leq 10) = \mathbb{P}(L \geq 20) = 0.22$. There is a $\boxed{22\%}$ probability of finding 20 or greater large bison, or equivalently of finding 10 or less small bison.

As an alternative to reformulating the question, R has a built-in way of dealing with the *greater-than-or-equal-to* case. If we specify the parameter value `lower.tail=FALSE`, it will add up from $k+1$ to n rather than from 0 to k. Hence, noting that $\mathbb{P}(L) = 1 - \mathbb{P}(S) = 0.58$, the command `pbinom(19,30,0.58,lower.tail=FALSE)` also yields the correct answer of 0.22. Note that for any k, `pbinom(k,n,p)` + `pbinom(k,n,p,lower.tail=FALSE)` = 1. As you may have guessed, the default value for the `lower.tail` parameter is `TRUE`.

What's Up With The Parameter `q`?

Notice that when using `dbinom()`, we used the parameter `x` to indicate the number of heads. However, when using `pbinom()`, we use the parameter `q` to indicate the number of heads. The reason we use a letter "q" is that cumulative probability is directly related to quantiles. Remember from Ch. (4) that a quantile q_p is the datum such that a fraction p of the values lie below it. This is similar to the way we use `q` in `pbinom()`, where we are interested in the finding the amount of total probability that lies below the value of `q`.

What If We Don't Want To Start at Zero?

The `pbinom()` function adds up all probabilities starting at $k = 0$ and ending at some specified $k = q$. What if we wanted to start adding probabilites at a non-zero value of k? For example, returning to our example of $n = 10$ flips of a $p = \mathbb{P}(H) = 0.7$ coin, how could we compute the probability of obtaining between 4 and 7 heads? Clearly this is just $\mathbb{P}(4) + \mathbb{P}(5) + \mathbb{P}(6) + \mathbb{P}(7)$. How could we do this this in R? The "brute force" method is to issue the command `sum(dbinom(x=4:7,size=10,prob=0.7))`. There is a more powerful method, however, that will be *much* more convenient when we study continuous probability distributions. Note that

$$\texttt{pbinom(7,10,0.7)} = \sum_{k=0}^{7} \mathbb{P}(k) = \mathbb{P}(0) + \mathbb{P}(1) + \mathbb{P}(2) + \cdots + \mathbb{P}(6) + \mathbb{P}(7)$$

and that

$$\texttt{pbinom(3,10,0.7)} = \sum_{i=0}^{3} \mathbb{P}(i) = \mathbb{P}(0) + \mathbb{P}(1) + \mathbb{P}(2) + \mathbb{P}(3).$$

If we *subtract* the equations, the terms $\mathbb{P}(0)$, $\mathbb{P}(1)$, $\mathbb{P}(2)$, and $\mathbb{P}(3)$ will cancel, and we'll be left with the terms we care about:

$$\begin{aligned}
\mathbb{P}(4 \leq k \leq 7) &= \sum_{k=4}^{7} \mathbb{P}(k) = \mathbb{P}(4) + \mathbb{P}(5) + \mathbb{P}(6) + \mathbb{P}(7) \\
&= \Big(\cancel{\mathbb{P}(0)} + \cancel{\mathbb{P}(1)} + \cancel{\mathbb{P}(2)} + \cancel{\mathbb{P}(3)} + \mathbb{P}(4) + \mathbb{P}(5) + \mathbb{P}(6) + \mathbb{P}(7) \Big) \\
&\quad - \Big(\cancel{\mathbb{P}(0)} + \cancel{\mathbb{P}(1)} + \cancel{\mathbb{P}(2)} + \cancel{\mathbb{P}(3)} \Big) \\
&= \texttt{pbinom(7,10,0.7)} - \texttt{pbinom(3,10,0.7)} \\
&\approx 0.607.
\end{aligned}$$

This is an idea we'll use frequently: to compute the probability of an outcome being in a certain range, we subtract the cumulative probability at the upper and lower limit of the range.

Example 13.5: Probability of Range of Heads

Problem: A coin with heads probability $p = 0.6$ is flipped 1000 times. What is the probability of obtaining betwen 575 and 625 heads?

Solution: Using the ideas described above, we can write that

$$\mathbb{P}\left(575 \leq k \leq 625\right) = \mathbb{P}(k \leq 625) - \mathbb{P}(k \leq 574)$$

Note that we specify the lower k as 574 instead of 575. The reason is that we want $\mathbb{P}(575)$ to be included in the answer, so we should only subtract off probabilites $\mathbb{P}(0)$ to $\mathbb{P}(574)$.

The R code

```
n <- 1000       # of flips
p <- 0.6        # heads probability
k.upper <- 625
k.lower <- 574
answer <- pbinom(q=k.upper,size=n,prob=p)
          - pbinom(q=k.lower,size=n,prob=p)
```

gives $\boxed{\mathbb{P}(575 \leq k \leq 625) = 0.90}$. For our $p = 0.6$ coin, there is a 90% probability we flip within 25 heads of the average number of 600.

13.5 The Poisson Distribution

A useful special case of the binomial distribution arises when considering occurrences/instances of events in some region of space or time. Let λ be the average # of events in some continuous interval I. That's a bit abstract, so here are some examples of what λ and I might represent:

- λ = average # of traffic accidents in $I = 1$ day
- λ = average # of text messages you receive in $I = 1$ hour
- λ = average # of genetic mutations happening every $I = 30$ minutes
- λ = average # of plants in $I = 5$ square meters of land
- λ = average # of coyote per $I = 1$ acre of land

Whenever we are dealing with an average # of independent occurrences/instances in some region of time or space, the `Poisson Probability Distribution`, or just *Poisson Distribution*, can be used to describe the situation. Before we give the equation, let's briefly discuss how this kind of situation relates to the binomial distribution.

How Is This Related To The Binomial Distribution?

At first glance the examples above don't seem to have anything to do with binary events. For example, if there are an average of $\lambda = 20$ traffic accidents per day, then on any given day we might have anywhere from 0 to 20 or more accidents. There are way more than two possible outcomes, and it seemingly makes no sense to use the binomial distribution when there are more than 2 possible outcomes.

The trick is to break the region I into a bunch of smaller regions. Instead of considering a whole day, let's consider individually each of the $86,400$ seconds that make up a day. Because there are 20 accidents per day, the probability of an accident during any given second is about $p = 20/86,400 \approx 0.0002315$. This is tiny. Thus, the probability of *two* traffic accidents in any given second, given by the product rule, is approximately zero. Hence, during our one second time period, the only plausible outcomes are either 0 or 1 traffic accidents. The outcomes for our little time interval are binary, like a coin-flip! An entire day is just a big succession of individual seconds (coin-flips). The probability of having 17 accidents in a day is just the probability of flipping our $p = 0.0002315$ coin $86,400$ times and getting

17 heads, where $\mathbb{P}(H) = p = 0.00023$. As an exercise, verify that this probability is 7.6%.

The Formula

We just discussed how we could re-interpret the situation by breaking our interval I of time (or space) into a huge number n of tiny intervals, each one basically a coin-flip for something happening or being present. Since on average λ events happen in the interval I, the probability of this event in one of the n sub-intervals is $p = \lambda/n$. We can then compute the probability of k total occurrences in our interval I by computing the binomial probability of flipping a $p = \lambda/n$ coin n times and getting k heads:

$$\begin{aligned}\mathbb{P}(k) &= \frac{n!}{k!(n-k)!} \cdot p^k \cdot (1-p)^{n-k} \\ &= \frac{n!}{k!(n-k)!} \cdot \left(\frac{\lambda}{n}\right)^k \cdot \left(1 - \left(\frac{\lambda}{n}\right)\right)^{n-k} \end{aligned} \tag{13.3}$$

This formula is valid as long as we make n large enough. If n is too small, then we're not breaking the interval up into tiny enough pieces, and there may be a sizable probability of more than one event per piece, meaning it's no longer binary. While the math is beyond the scope of this book, it turns that if you let n get bigger and bigger, Eq. (13.3) settles on a single number that doesn't depend on n anymore. The final formula for the Poisson Distribution is

$$\mathbb{P}(k) \approx \frac{\lambda^k e^{-\lambda}}{k!} \tag{13.4}$$

which, believe it or not, is equivalent to Eq. (13.3) if you let n be big enough. For our traffic accident scenario, where there are $\lambda = 20$ accidents per day on average, the probability of 17 accidents in a day is

$$\mathbb{P}(17) = \frac{20^{17} e^{-20}}{17!} \approx 7.6\%$$

which can be quickly computed in R via the command

(R.13.4)
```
dpois(x=17,lambda=20)
```

The Poisson distribution for $\lambda = 20$ is shown in Fig. 13.3. The built-in R function for the cumulative probability is `ppois()`.

Remember that the symbol "e" is a mathematical constant with value $e \approx 2.7182828459$. R can evaluate e^x via the built-in function `exp()`. Hence, to compute e^{-20}, type `exp(-20)`.

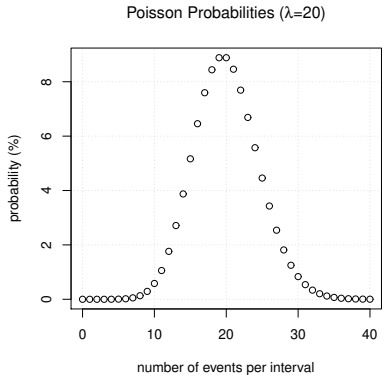


Figure 13.3: Plot of Poisson probabilities for $\lambda = 20$ using `dpois()`.

Example 13.6: Poisson Distribution

Problem: In a certain animal population, there are on average 15 births per month. What is the probability that there are between 13 and 17 births in a month?

Solution: Because we are dealing with an average number of events/occurrences (15 births) in an interval of time or space (1 month), assuming these events are *indpendent* we can use the poisson distribution. We start by storing our given information:

```
lambda <- 15      # number of births per month
k.up <- 17        # upper limit on # of births
k.low <- 13       # lower limit on # of births
```

We can now proceed in two different (but equivalent ways). We could compute `dpois()` for a vector of k values between 13 and 17, and then add up the resulting probabilities:

Note that we could collapse lines 1 thru 5 into the single command `sum(dpois(13:17,15))`, though for scripts it is better to lay out the details in separate lines.

```
prob.vals <- dpois(x=k.low:k.up,lambda=lambda)
answer <- sum(prob.vals)
```

which gives a probability of $\boxed{0.48}$. Alternatively, we could use cumulative probability functions:

Note the lower cumulative function goes up to `k.low-1` and not `k.low`, since we want to subtract off the probabilities for zero through 12 from the probabilities for zero through 17 in order to leave 13 through 17.

```
answer <- ppois(x=k.up,lambda=lambda)
          - ppois(x=k.low-1,lambda=lambda)
```

which also gives a probability $\mathbb{P}(13 \leq k \leq 17) = 0.48$.

13.6 Exercises

1. Consider a binomial experiment with $n = 30$ and $p = 0.3$, where the number of "successes" or "heads" is given by k. Write an R script that does the following:

   i) Computes $\mathbb{P}(k = 5)$.

   ii) Computes $\mathbb{P}(k \geq 7)$ using an explicit summation of `dbinom()`.

   iii) Computes $\mathbb{P}(k \geq 7)$ using the appropriate cumulative distribution function.

   iv) Computes $\mathbb{P}(5 \leq k \leq 21)$ via subtracting the appropriate cumulative distribution functions.

   v) Generates a plot of the appropriate binomial probability distribution.

   vi) Generates a plot of the appropriate cumulative probability distribution.

2. Consider flipping a fair coin 10 times, and denoting the number of heads you got by the symbol h_1, where the the "1" means this is the first time you do the experiment. Now repeat the experiment a second time: flip the coin 10 times, and denote the number of heads you got by h_2. Now imagine you keep repeating the experiment until you've done it 1,000,000 times. Thus, you'll have 1,000,000 values of h, the number of heads you got in each trial of the experiment.

   i) Out the million trials/experiments, how many of them would you expect to have 0 heads? Call this number $f(0)$, the expected frequency of 0 heads. How many of them would you expect to have 1 head? Call this number $f(1)$. And so on. Compute $f(0), f(1), f(2), \cdots, f(9), f(10)$.

   ii) Divide all of your f values by $N = 1,000,000$, and denote the results by P.

iii) Make a plot of P vs. the number of heads. Make a separate plot of the binomial distribution for 10 flips with p=0.5. Give a detailed explanation of the relationship between these two plots.

3. An ecological survey involves the partitioning of a field into *quadrats*. On average, each quadrat contains 4 plants.

   i) What probability distribution best describes this situation? Explain.

   ii) What is the probability that a randomly selected quadrat would have 2 plants? 8 plants?

   iii) If you examine 200 quadrats, how many quadrats would you expect to contain 2 plants? 8 plants?

   iv) Make a labeled plot of the function $\mathbb{P}(x)$, where $\mathbb{P}$ is the probability of finding x plants in a randomly selected quadrat.

4. You are told that 5% of a local population of plants is infected with a disease. You seek to verify the accuracy of this claim. You did your best to randomly select 40 plants, and you tested them to find out that 4 of the plants are diseased.

   i) If 5% of plants are infected, on average how many plants out of 40 would you expect to be infected? Call your answer $\bar{k}$.

   ii) Assuming the 5% disease rate, what is the probability $\mathbb{P}(4)$ of finding 4 infected plants? How does this number compare to $\mathbb{P}(\bar{k})$? Does your data seem compatible with the 5% number? In light of this, what do you think about the claim that the infection rate is 5%?

   iii) What is the probability of finding $\mathbb{P}(\geq 4)$ infected plants? In light of this number, what do you think about the claim that the infection rate is 5%?

   iv) Which analysis do you think offers a better ability to evaluate the correctness of the disease rate: analysis ii), or analysis iii)?

CHAPTER 14

Probability Densities

Most of the time quantitative data will be continuous rather than discrete. In this chapter we discuss how the idea of the probability distribution is applied to continuous data. To better motivate these ideas, we begin with a simplified model of measurement noise.

14.1 A Simplified Model of Measurement Noise

Recall from Sec. 8.1 that instead of measuring the "true" average value of a quantity, we actually measure the result of the "true" value modified by a bunch of different un-measured factors that push the true value around on the number line. The net result of all these pushes is called *noise*. We illustrated this process for a few different factors in Fig. 8.10, reproduced below.

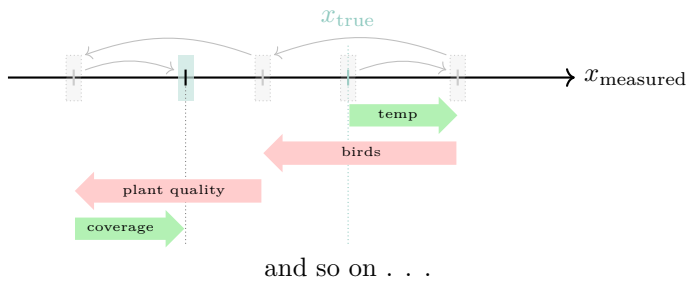


Figure 14.1: Here x_{true} is the true average number of insects munching on plants receiving a certain pesticide. As a number of different factors we don't measure bump the value x_{measured} left and right, the measured value x_{measured} bounces around on the number line, eventually landing at its final measured value. All of these variations in x_{measured} result in *noise* in the data.

Instead of considering the pesticide example from Sec. 8.1, let's consider a plant population with a true average height $h_{\text{true}} = 67\ cm$. We will examine a simplified model of noise in which each of the influencing factors is modeled as a binomial event. Assume there are 50 such factors that would result in measurable differences in height. We'll assume each factor that could influence the plant's height has a 50% chance of increasing height by a fixed amount $h_{\text{bump}} = 0.2\ cm$, and a 50% chance of decreasing height by this same amount. While this model is too simplified to be realistic, it will illustrate the underlying idea of how we can quantify noise.

Computing A Probability Distribution for Height

Because there are $n = 50$ influencing factors, each of which is a binomial event, we arrive at the measured height by starting with the "true" height $h_{\text{true}} = 67\ cm$ and

performing $n = 50$ coin-flips, where each heads would add $h_{\text{bump}} = 0.2\ cm$ of height, and each tails would subtract $h_{\text{bump}} = 0.2\ cm$ of height. If *all* of the flips landed heads, meaning each influencing factor ended up favoring an increased height, then the measured height would be $h_{\text{measured}} = h_{\text{true}} + 50 \times 0.2 = 67 + 10 = 77\ cm$. On the other hand, if all 50 flips discouraged growth, the measured height would be $h_{\text{measured}} = h_{\text{true}} - 50 \times 0.2 = 57\ cm$.

In general, if k of the n flips land heads, we would have a measured height

$$\begin{aligned} h_{\text{measured}} &= h_{\text{true}} + h_{\text{bump}} \times (\#\text{ of heads}) - h_{\text{bump}} \times (\#\text{ of tails}) \\ &= h_{\text{true}} + h_{\text{bump}} \cdot k - h_{\text{bump}} \cdot (n-k) \\ &= 67 + 0.2k - 0.2(n-k)\ \ cm \end{aligned} \tag{14.1}$$

This formula gives the measured height when k of the influencing factors favor growth (heads). For example, if $k = 23$, we have $h_{\text{measured}} = 67 + 0.2(23) - 0.2(50 - 23) = 66.2\ cm$. This number is a bit less than $h_{\text{true}} = 67$ since there were a few more tails than heads, meaning there were more factors that suppressed height than there were factors that enhanced height.

We can compute the probability of k heads in n flips using the binomial distribution Eq. (13.1), which is also the probability of obtaining the h_{measured} that corresponds to this number of heads. For example, because the probability of $k = 23$ heads is `dbinom(23,50,0.5)` $= 9.6\%$, the probability of $h_{\text{measured}} = 66.2\ cm$ is thus also 9.6%.

Plot The Results in R

The following R-code creates a graph of the probability of various values of h_{measured} in our simplified model.

```
n <- 50             # number of influencing factors
p <- 0.5            # probability of heads/tails
k <- 0:50           # vector of possible # of heads
h.true <- 67        # true average height in cm
h.bump <- 0.2       # amount each factor bumps height
h.measured <- h.true + h.bump*k - h.bump*(n-k)
prob.vals <- dbinom(x=k,size=n,prob=p)
plot(x=h.measured,
     y=prob.vals,
     main="Probability Distribution of Height",
     xlab="height (cm)",
     ylab="probability",
     xlim=c(60,74),
     panel.first=grid())
```

(R.14.1)

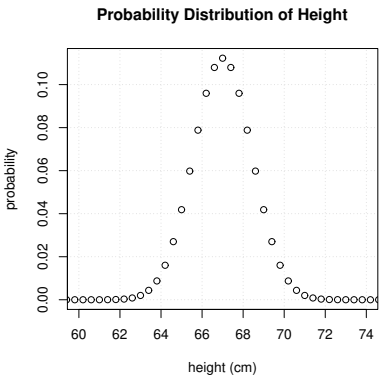


Figure 14.2: Probability Distribution for h with $n = 50$ binomial factors that can influence the measurement. If the number of factors gets very large, the probability distribution will become (approximately) a continuous curve rather than a discrete set of points.

The resulting probability distribution for h_{measured} is shown in Fig. 14.2. Notice that the most likely possibility is the true value of 67 cm, but nearby values are almost as probable. You could very easily measure 69 cm, for example. Most of the time we'll get something in the vicinty of 67, where "the vicinity" means something like a couple cm here. Furthermore, there is an off chance we measure something really far away, like 62. When reporting a measurement, our goal will be to quantify a) the size of "vicinity", and b) indicate how probable it is that we landed in it. Quantifying both of these things will rely upon the precise details of the probability distribution in question.

Limitations In This Simplified Model

Our simplified model of noise has a number of over-simplifications.

1. We assumed each influencing factor either adds or subtracts a fixed amount. Typically each factor actually adds or subtracts a varying amount depending on its value and the type of factor it is.
2. It's not the case that all of the influencing factors *independently* bump the value of x. How much one factor bumps x will often depend upon the value of the other factors.
3. There will often be far more than 50 influencing factors.

The result of all this is that typically something like height will have a *continuous* number of possibilities, rather than discrete. We need to assign probabilities to a continuous (infinite) number of possible outcomes. In real situations, noise is better modeled by various types of *continuous* probability distributions.

14.2 From Discrete to Continuous

The binomial and Poisson distributions are examples of discrete distributions, in the sense that we are assigning probabilities to discrete numbers: $\mathbb{P}(4 \text{ heads})$ makes sense, but $\mathbb{P}(3.52 \text{ heads})$ doesn't. However, many types of data we collect are continuous. An example is the height h of a person, which could be any decimal between the minimum and maximum height of a human. Somebody could be $h = 1.03456\ldots$ meters, $h = 1.78250\ldots$ meters, etc. Somehow we need to assign probabilities to an *infinite* number of possible heights.

Remember that a probability distribution is a rule for how the total probability of 1 is spread out among the possibilities. Thus, it would appear that a probability distribution for height should assign a probability to each height value. However, this isn't a useful point of view, because if there are truly an infinite number of possible heights, then the probability of any one particular height must be zero! For example, it is clear that the probability of finding somebody with a height of exactly $h = 2.140124005092039010...$ meters is zero. On the other hand, it certainly makes sense to ask for the probability that somebody has a height between 2.14 and 2.15 meters. An equation that gives the probability for a continuous variable to be in some interval is known as a `probability density function` or pdf for short, often symbolized by the Greek letter ρ. *Probability densities assign probabilities to a continuous range of numbers, and NOT to a specific number.*

For a continuous probability distribution, a finite amount of probability doesn't exist at a single point, but instead is spread out over a range of values. This is similar to how a finite amount of mass can't exist at a singular point, but can be spread out over a region of space, resulting in some average density of matter. It is in light of this analogy that a continuous probability distribution function is called a `probability density function`.

Thus, sticking with our height example, something like $\mathbb{P}(h = 1.79)$ is zero, but $\mathbb{P}(1.78 < h < 1.79)$ is finite. When dealing with a probability density function, we are *always* dealing with probabilities of intervals, NOT probabilities of a single number.

14.3 Probability = Area Under Curve

In the graph of a discrete probability distribution, the probability of getting a certain value is given by the y-coordinate of the graph. In a continuous probability density, to get a probability that a value is in some interval, you compute the area under the curve within this interval. The idea is illustrated by the following example.

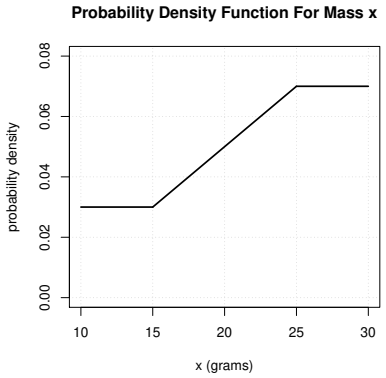


Figure 14.3: Probability density function for the weight x of some object.

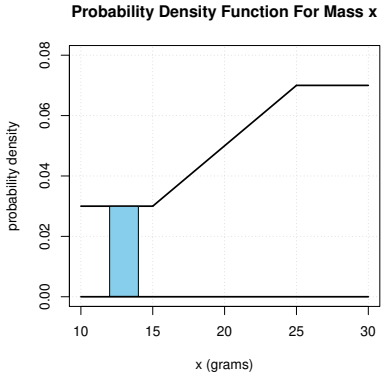


Figure 14.4: $\mathbb{P}(12 < x < 14)$ = shaded area under curve = area of rectangle.

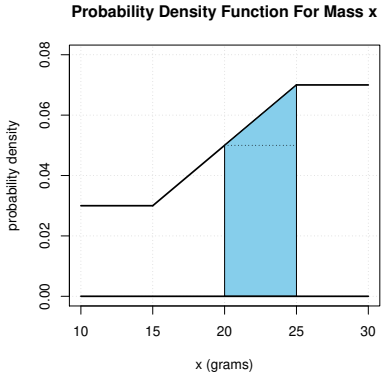


Figure 14.5: $\mathbb{P}(20 < x < 25)$ = shaded area under curve = area of rectangle + triangle.

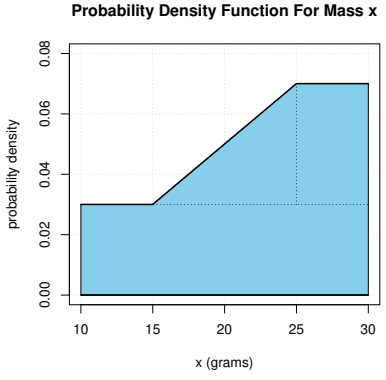


Figure 14.6: $\mathbb{P}(10 < x < 30)$ = shaded area under entire curve = area of 2 rectangles + 1 triangle.

Example 14.1: Probability Is Area

Problem: Assume that the mass (in grams) of a certain type of object has the probability density shown in Fig. 14.3. For a randomly selected object, compute the probability that the mass x is:

i) between 12 and 14 grams.

ii) between 20 and 25 grams.

iii) between 10 and 30 grams.

iv) exactly 14 grams.

Solution: Each of the above probabilities is given by the corresponding area under the probability density curve. Let's look at each probability in turn:

i) $\mathbb{P}(12 < x < 14)$ is given by the area under the density curve between $x = 12$ and $x = 14$. As shown in Fig. 14.4, the corresponding area under the curve is a rectangle. The width of this rectangle is $14 - 12 = 2$, and the height of this rectangle looks to be about 0.03. The area of a rectangle is width times height, so the area we want is $2 \times 0.03 = 0.06$. Thus, $\boxed{\mathbb{P}(12 < x < 14) = 0.06}$.

ii) $\mathbb{P}(20 < x < 25)$ is the area of the shaded region shown in Fig. 14.5. This region can be thought of as a rectangle with a triangle sitting on top. The area of the rectangle is $w \times h = 5 \times 0.05 = 0.25$ and the area of the triangle is $\frac{1}{2}b \times h = \frac{1}{2} \times 5 \times (0.07 - 0.05) = 0.05$. The total area is $0.25 + 0.05 = 0.3$, and hence $\boxed{\mathbb{P}(20 < x < 25) = 0.3}$.

iii) $\mathbb{P}(10 < x < 40)$ is the area of the region beneath the entire curve, as shown in Fig. 14.6. This region is comprised of two rectangles and a triangle. The bottom (larger) rectangle has area $w \times h = 20 \times 0.03 = 0.6$. The top (smaller) rectangle has area $w \times h = 5 \times 0.04 = 0.2$. The triangle has area $\frac{1}{2}b \times h = \frac{1}{2} \times 10 \times 0.04 = 0.2$. The sum of these areas is $0.6 + 0.2 + 0.2 = 1$. Thus, $\boxed{\mathbb{P}(10 < x < 30) = 1}$.

From this we conclude that the entire density curve is shown: because a total probability of 1 is contained within the shaded region, any values of x less than 10 or greater than 30 have zero probability. All the possible values of x are shown in the graph.

iv) There is no area underneath a single point. Hence, $\boxed{\mathbb{P}(x = 14) = 0}$. One way of visualizing this is to note that we are computing the area under the curve (where the density $\rho = 0.03$) of a region between $x = 14$ and $x = 14$, which is a rectangle of width=0.

As discussed in Sec. 14.2, when there are an infinite number of possible values of x (any decimal between 10 and 30), the total probability of 1 is spread out among an infinite number of possibilities, and hence the probability of any specific value is zero. When we say $x = 14$, we're really saying $x = 14.00000000000000...$, and the probability of being *exactly* that value is zero. It only makes sense to compute probabilities in some interval. For example, it makes sense to compute $\mathbb{P}(13.9 < x < 14.1)$, etc.

While this example does drive home the idea that probabilities are areas, most of the time probability density curves are actual *curves*, meaning they aren't made of straight-line segments. Thus, in general, you can't find areas by adding up a few rectangles or triangles. Luckily, R can easily compute the area beneath most types of probability densities.

Even for a curvy curve, however, one can approximate the area by filling the region with a very large number of really thin rectangles that fit under the curve, and then adding up all the rectangles. This general procedure is a central operation in calculus. We don't, however, assume knowledge of calculus in this book.

14.4 The Normal Density

In a situation where there are a huge number of influencing factors that either add or subtract some amount from the true value, the noise is usually described by the normal density. It is the most important density in statistics.

14.4.1 General Features

The `normal density`, also known as the Gaussian density, is a probability density function with the shape of bell. The shape of this curve is determined entirely by two parameters: the mean μ and the standard deviation σ. Once these two numbers are specified, one can compute the probability for a quantity to lie within some specified range. The particular case of a normal density with $\mu = 50$ cm and $\sigma = 10$ cm (representing, say, the height of an animal) is shown in Fig. 14.7.

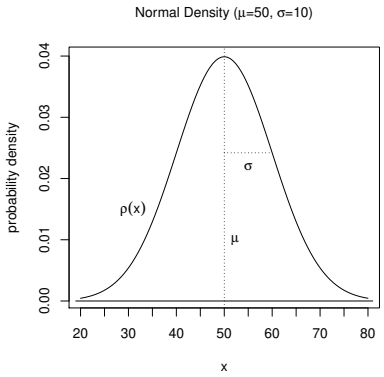


Figure 14.7: Plot of Normal Probability Density Function for $\mu = 50, \sigma = 10$.

The plot only shows $20 \leq x \leq 80$, though the normal curve actually extends from $-\infty$ to $+\infty$.

Recalling our discussions of noise, evidently μ represents the "true" average value of the quantity. Because there are a huge number of influencing factors that push x up and down by various amounts, we typically won't actually measure $x = \mu$, but rather some x in the vicinity of μ. The size of this vicinity is roughly given by $\pm\sigma$. The standard deviation σ thus relates to the amount of noise typically present. When a normal density has a very large σ, this means the noise is large, and we may easily find an x that is pretty far away from the true average of μ. Conversely, when σ is very tiny the noise level is pretty low, and we'll usually find an x that is pretty close to μ.

Normal densities actually extend all the way to $\pm\infty$, though the density ρ (i.e. the height of the curve) becomes very close to zero when you go out far enough. Still, mathematically speaking, there is a finite probability of getting arbitrarily negative or positive values. Clearly this doesn't make sense in most physical situations (e.g. you couldn't have a negative height or weight), and thus the normal density is at best a very good approximation in most real situations.

The R function `dnorm()` gives the value of the normal density ρ. However, we usually aren't interested in the height of the density curve, but rather the area under the curve, i.e. a probability. Hence, most of the time we will make use of the `pnorm()` function, which gives the total area from $-\infty$ up to a specified value.

There is an exact formula for the normal density:

$$\rho(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

In this book we won't use this formula, though, as we'll always just use `dnorm()`. Furthermore, there is no exact formula for the cumulative probability, so we're forced to use something like `pnorm()`, which provides a very good decimal approximation.

14.4.2 Computing Probabilities

Consider the distribution shown in Fig 14.7. The probability of finding an animal with a height between 60 and 70 cm is then equal to the area underneath the density curve between 60 and 70, as indicated by the shaded region in Fig. 14.8. This area is given by

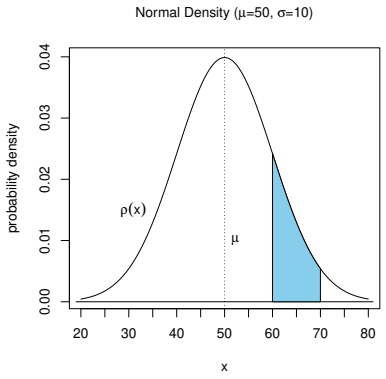


Figure 14.8: Area indicating probability for $60 \leq x \leq 70$.

$$\begin{aligned}
\mathbb{P}(60 \leq x \leq 70) &= \text{area between 60 and 70} \\
&= \\
&= \Big(\text{area between } -\infty \text{ and } 70\Big) - \Big(\text{area between } -\infty \text{ and } 60\Big) \\
&= \quad - \\
&= \texttt{pnorm(70,mean=50,sd=10)} - \texttt{pnorm(60,mean=50,sd=10)} \\
&= \texttt{pnorm(70,50,10)} - \texttt{pnorm(60,50,10)} \\
&\approx 0.136
\end{aligned} \tag{14.2}$$

A useful R function for probability densities is the quantile function. For the normal density, this function is called `qnorm()`. You feed this function a probability p, and it returns a quantile, i.e. the value of x such that there is a total area of p below this value.

Example 14.2: Using `qnorm()`

Problem: A normal density has $\mu = 30$ and $\sigma = 4$. 70% of randomly selected values would be less than or equal to what value of x?

Solution: We are asked to find the 70^{th} percentile of the density. This is computed in R as

```
qnorm(p=0.7,mean=30,sd=4)
```

with result 32.1. Thus, $\boxed{\mathbb{P}(x < 32.1) \approx 0.7}$.

We can verify this result by computing `pnorm(32.1,mean=30,sd=4)`, which returns 0.7. In this way, we see that `pnorm()` and `qnorm()` are, in the language of algebra, inverse functions.

14.4.3 The Standard Normal Density

It turns out that *every* normal density is identical if each value of x is expressed by how many standard deviations it is from the mean. Consider our previous example of $\mu = 50$, $\sigma = 10$. The value $x = 40$ is one standard deviation to the left of the mean. The value $x = 65$ is 15 units (i.e. 1.5 standard deviations) to the right of the mean. This measure of standard deviations from the mean is known as the `standard score` of the variable, and is denoted z in the case of the normal density. In equation form, the standard score z of a value x, a.k.a. the `z-score` is

$$z \equiv \frac{x - \mu}{\sigma}. \tag{14.3}$$

For the case of $x = 40$ when $\mu = 50$, $\sigma = 10$, we have $z = (40 - 50)/10 = -1$, meaning that our value of 40 is one standard deviation to the left of μ. On the other hand, for $x = 65$ we have $z = (65 - 50)/10 = +1.5$, meaning the value of 65 is 1.5 standard deviations to the right of the mean.

If the random variable x is expressed in terms of standard-scores then every normal density takes the shape shown in Fig. 14.9.

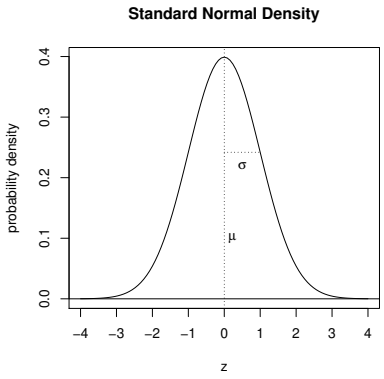


Figure 14.9: Standard normal probability density. Note that when expressing values as standard scores, the mean μ is always $z = 0$, and the width of the distribution is always $\sigma = 1$.

Example 14.3: Using z-Scores

Problem: In Eq. (14.2) we computed $\mathbb{P}(60 < x < 70)$ for a normal density with mean $\mu = 50$ and standard deviation $\sigma = 10$. Repeat this calculation using standard scores.

Solution: First we express the limiting values of $x = 60$ and $x = 70$ in z-scores:

$$x = 60 \quad \rightarrow \quad z = \frac{60 - 50}{10} = 1$$
$$x = 70 \quad \rightarrow \quad z = \frac{70 - 50}{10} = 2$$

We are thus looking for the probability $\mathbb{P}(1 \leq z \leq 2)$, i.e. that our random variable is between +1 and +2 standard deviations from the mean. This probability is given by the shaded area in Fig. 14.10. In R the default values of μ and σ are 0 and 1 in `pnorm()`, and thus when using the standard normal density the only argument required for `pnorm()` is the z-score:

$$\begin{aligned}
\mathbb{P}(1 \leq z \leq 2) &= \text{area between 1 and 2} \\
&= \quad \text{ } \\
&= \Big(\text{area between } -\infty \text{ and } 2\Big) - \Big(\text{area between } -\infty \text{ and } 1\Big) \\
&= \quad \text{ } \\
&= \texttt{pnorm(2)} - \texttt{pnorm(1)} \\
&\approx 0.136 \qquad (14.4)
\end{aligned}$$

which is the same probability obtained in Eq.(14.2).

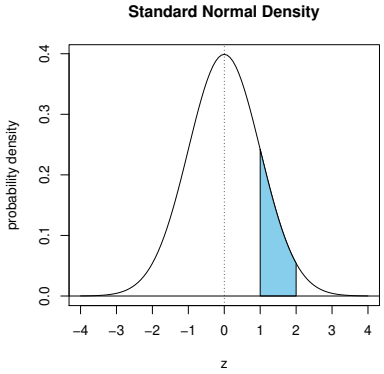


Figure 14.10: Area indicating probability for the standard score to lie in the range $1 \leq z \leq 2$, which is equivalent to the original variable being in the range $60 \leq x \leq 70$.

When dealing with normal densities, we will often convert the random variables to z-scores and deal with the standard normal density.

14.5 Some Other Densities We'll Use

While the normal density is the most prominent probability density in statistics, there are a very large number of different types of densities that show up in different situations. Examining most of these is beyond the scope of the book, but here are three more that do show up later on in the text. Each of the three is ultimately related to the normal density.

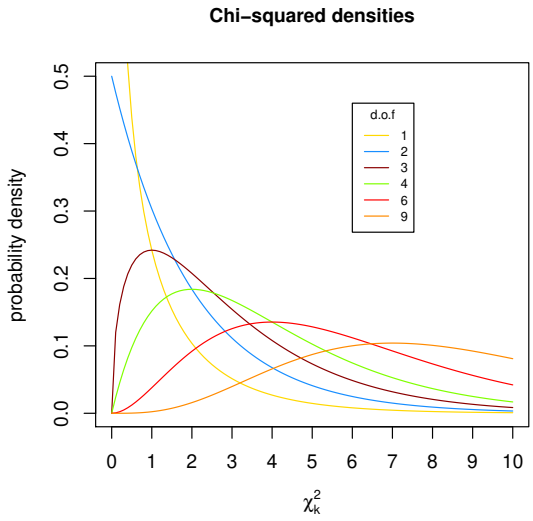


Figure 14.11: χ^2 distribution for different d.o.f. k.

The chi-squared density describes, for example, the probability density of sample variances one obtains from measurments of a normal density. It also arises when assessing how well a data-set seems to fit various assumed probability densities.

14.5.1 Chi-squared Density

If instead of looking at a normal random variable z you look at the *square* of that random variable, z^2, then the resulting probability density is known as a `$\chi^2$ (or chi-squared) density` with one `degree-of-freedom` (i.e. you're looking at the square of one standard normal variable z).

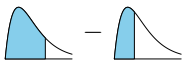
One could also look at a sum of squares of multiple standard normal variables $z_1, z_2, \ldots$, where z_1 and z_2 etc. are different random variables and *not* simply subscripted data as discussed in Sec. 1.4.1. If one considers the sum of k squares of standard normal random variables, the resulting random variable follows a χ^2 density with k degrees of freedom:

$$\chi^2_k = \sum_{i=1}^{k} z_i^2 = z_1^2 + z_2^2 + \cdots + z_k^2 \tag{14.5}$$

The shape of the probability density function is different for different degrees of freedom (d.o.f.) k, as shown in Fig. 14.11. Note that because the square of a number is always positive, the random variable χ^2_k must always be positive, and thus the density curves start at $\chi^2_k = 0$. It is also important to note that the χ^2_k density becomes ever *flatter* as k increases. For very large values of k, the χ^2_k density is pretty close to a flat horizontal line. This fact will be of relevance when discussing the t-density in the next section.

As in the case of the normal density, probabilities are given by areas. Consider the specific case of d.o.f.$= k = 4$. The probability that χ^2_4 lies between 3 and 6 is the area under the χ^2_4 density curve between 3 and 6, as shown in Fig. 14.12. To

compute this area we use the `pchisq()` function in R:

$$
\begin{aligned}
\mathbb{P}\left(3 \le \chi^2_4 \le 6\right) &= \text{area between 3 and 6} \\
&= \quad \text{ } \\
&= \Big(\text{area between 0 and 6}\Big) - \Big(\text{area between 0 and 3}\Big) \\
&= \quad \text{  } \\
&= \texttt{pchisq(6,df=4)} - \texttt{pchisq(3,df=4)} \\
&= \texttt{pchisq(6,4)} - \texttt{pchisq(3,4)} \\
&\approx 0.359
\end{aligned} \tag{14.6}
$$

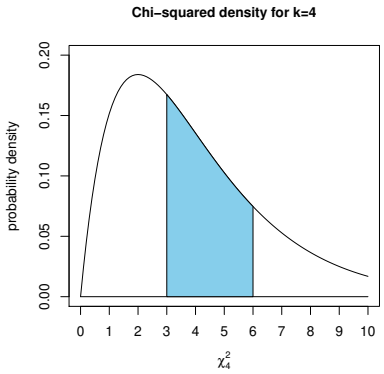


Figure 14.12: Area indicating probability for $3 \le \chi^2_4 \le 6$.

14.5.2 t Density

One common usage of the t-distribution is modeling the mean of a small sample of measurements from a normal distribution.

The `t-density` arises when one considers the ratio of z and χ^2_k random variables that are independent of each other. Symbolically,

$$t = \frac{z}{\sqrt{\chi^2_k / k}} \tag{14.7}$$

Note that the t random variable also depends upon the d.o.f. k of the χ^2_k density. Thus, the shape of the t-density will depend upon the d.o.f. k, as shown in Fig. 14.13. As mentioned in Sec. 14.5.1, when the degrees of freedom k becomes large enough, the χ^2_k density becomes more-or-less constant, and Eq. (14.7) implies that the t-density becomes very close to a z density. Thus, *for large degrees of freedom k, the t-density becomes a standard normal density.*

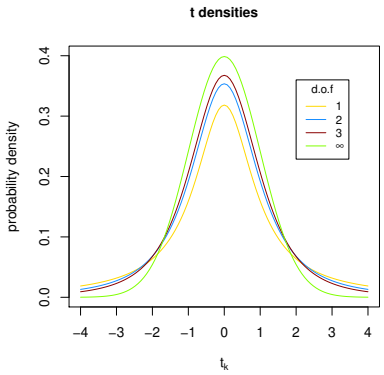


Figure 14.13: t-density for different d.o.f. k.

How large does k need to be in order to use a z density in place of the t density? The question makes no sense if you don't specify the accuracy required in one's calculation. Is it good enough to be within 5%? 1%? 0.001%? While there is a commonly used rule of thumb that $k \approx 30$ is sufficient, such rules of thumb make no sense if you don't specify a desired accuracy. Later we will show how to give more precise answers to this question through the use of simulations in R.

As in the other continuous distributions discussed thus far, probabilities are given by areas underneath the probability density curve. For example, what is the probability that $-2 \le t \le 1$ for a t-distributed random variable with degrees of freedom $k = 7$? This probability $\mathbb{P}(-2 \le t \le 1)$ is given by the area underneath the $k = 4$ t-density curve between -2 and 1, as shown in Fig. 14.14. This probability is

$$
\begin{aligned}
\mathbb{P}(-2 \le t \le 1) &= \text{area between } -2 \text{ and } 1 \\
&= \quad \text{ } \\
&= \Big(\text{area between } -\infty \text{ and } 1\Big) - \Big(\text{area between } -\infty \text{ and } -2\Big) \\
&= \quad \text{ } \\
&= \texttt{pt(1,df=7)} - \texttt{pt(-2,df=7)} \\
&= \texttt{pt(1,7)} - \texttt{pt(-2,7)} \\
&\approx 0.782
\end{aligned} \tag{14.8}
$$

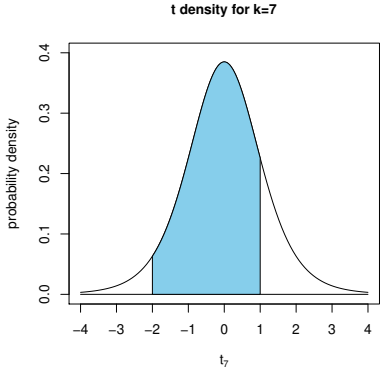


Figure 14.14: Area indicating probability for $-2 \le t \le 1$.

One application of the F-density is in modeling the underlying difference in groups of data via the ANOVA method.

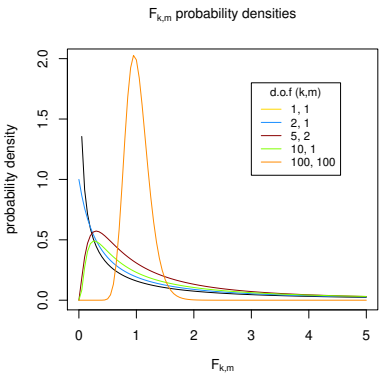


Figure 14.15: F-density for different degrees of freedom k, m

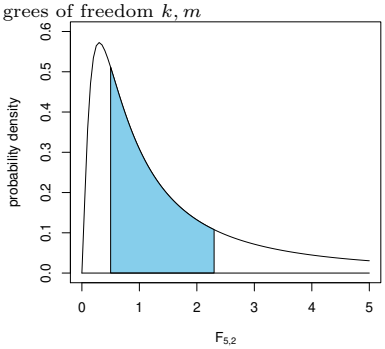


Figure 14.16: Area indicating probability for $0.5 \leq F_{5,2} \leq 2.3$.

14.5.3 F Density

The last continuous distribution we will mention is the `F-density` , which is the ratio of two independent χ^2 densities that have each been scaled by their degrees of freedom. Symbolically,

$$F_{k,m} = \frac{\chi^2_k/k}{\chi^2_m/m} \tag{14.9}$$

where it is explicit that the F-density depends upon the degrees of freedom k and m of the two χ^2 densities, as shown in Fig. 14.15.

The probability that an F random variable lies in a certain range is, as usual, given by the area under the $F_{k,m}$ curve between the two values. For the case of an F-density with degrees of freedom $k, m = 5, 2$, the probability of F lying between 0.5 and 2.3, i.e. $\mathbb{P}\left(0.5 \leq F_{5,2} \leq 2.3\right)$, is equal to the shaded area in Fig. 14.16

$$\begin{aligned}
\mathbb{P}\left(0.5 \leq F_{5,2} \leq 2.3\right) &= \text{area between 0.5 and 2.3} \\
&= \\
&= \Big(\text{area between 0 and 2.3}\Big) - \Big(\text{area between 0 and 0.5}\Big) \\
&= \quad - \\
&= \texttt{pf(2.3,df1=5,df2=2)} - \texttt{pf(0.5,df1=5,df2=2)} \\
&= \texttt{pf(2.3,5,2)} - \texttt{pf(0.5,5,2)} \\
&\approx 0.440
\end{aligned} \tag{14.10}$$

14.6 Exercises

Figure 14.17: A female Monarch butterfly.

1. A entomology textbook reports that the monarch butterfly (*Danaus plexippus*) has a mean weight of $\bar{w} = 1.775 \cdot 10^{-2}$ g, with a standard deviation $\sigma = 8.25 \cdot 10^{-3}$ g. Assume w follows a normal density.

   i) Compute the standard scores of $w_1 = 2.13 \cdot 10^{-2}$ g and $w_2 = 2.45 \cdot 10^{-2}$ g.

   ii) Use the standard normal density to compute the probability that a random butterfly has a weight w between w_1 and w_2. Draw a standard normal curve and shade the area corresponding to this probability.

   iii) What is the probability that a random butterfly has a weight that is at least 3 σ above the mean?

   iv) While it may make sense to approximate the weight density with a normal density, the actual exact weight density *cannot* be a true normal density. Explain why.

v) Imagine that you conduct a research study of 250 Monarch butterflies, and you find a mean weight of $1.56 \cdot 10^{-2}$ g and a standard deviation of $1.31 \cdot 10^{-2}$ g. Given your results, what is your assessment of the figures reported in the entomology textbook?

2. Compute the probability that a random variable x has a value $1 \leq x \leq 2$ assuming that x is described by a:

   i) standard normal density

   ii) t-density with degree of freedom $k = 5$

   iii) χ^2 density degree of freedom $k = 7$

   iv) F-density with degrees of freedom $k = 3$ and $m = 7$

3. A normal density has mean $\mu = 50$ and $\sigma = 10$.

   i) Find the number X such that 30% of the probability lies below this x, i.e. $\mathbb{P}(x < X) = 0.3$.

   ii) Find the interval centered around μ that contains 85% of the probability. In other words, find the number y such that

   $$\mathbb{P}(\mu - y \leq x \leq \mu + y) = 0.85$$

   iii) Find left-tailed region that contains 85% of the probability. In other words, find the number y such that

   $$\mathbb{P}(-\infty \leq x \leq y) = 0.85$$

   iv) Find the double-tailed region containing 85% of the probability. In other words, find the number y such that

   $$\mathbb{P}(-\infty \leq x \leq \mu - y) + \mathbb{P}(\mu + y \leq x \leq \infty) = 0.85$$

   For each of these cases, in addition to providing the R-command(s) and results, please sketch a normal density with the appropriate shaded area. Also express each y as a standard score (i.e. as a number of standard deviations σ).

Part III

Bayesian Inference

CHAPTER 15

Bayesian Inference From Binomial Data

Having introduced descriptive statistics and probability theory in Parts I and II, we are now ready for our real task: to draw inferences or form conclusions on the basis of data. In this chapter we'll discuss how Bayesian methods can help us analyze situations with binomial data.

15.1 Formulating the Question & Answers

The sequence of steps in our analysis will be as follows:

1. Clearly formulate the question.
2. Organize data.
3. Specify statistical models.
4. Compute posteriors with Bayes' Theorem.
5. Analyze the posterior and compute credible intervals.

Note that from the perspective of *designing* an experiment, you'll want to plan out all parts of the analysis *before* you collect data.

First we need a clear question, which also implies we have in mind the general form of possible answers. For example, let's assume we're investigating a coin and are trying to deduce the *the probability of flipping heads*, denoted p. A clear scientific question is:

What is the value of p for the coin?

With the question clearly stated, one can then formulate possible answers. One possible answer to the question is that p lies between 0.45 and 0.55:

$0.45 < p < 0.55$ or equivalently $p \in (0.45, 0.55)$.

Such a possible answer to our question is known as a `hypothesis`, and there is always more than one hypothesis (i.e. possible answer to our question). We will denote the i^{th} hypothesis with the symbol H_i. For our question, we could thus consider the following set of 20 hypotheses:

$$\begin{array}{rl} H_1: & p \in (0.00, 0.05) \\ H_2: & p \in (0.05, 0.10) \\ H_3: & p \in (0.10, 0.15) \\ \vdots & \\ H_{19}: & p \in (0.90, 0.95) \\ H_{20}: & p \in (0.95, 1.00) \end{array}$$

There are other possible sets of answers to our question. We could have intervals that are bigger or smaller than 0.05. For example, intervals of size 0.01 yield 100 distinct hypotheses:

$$\begin{array}{rl} H_1: & p \in (0.00, 0.01) \\ H_2: & p \in (0.01, 0.02) \\ H_3: & p \in (0.02, 0.03) \\ \vdots & \\ H_{99}: & p \in (0.98, 0.99) \\ H_{100}: & p \in (0.99, 1.00) \end{array}$$

We can state the above 100 hypotheses more compactly as

$$H_i : p \in \Big(0.01 \cdot (i-1)\ ,\ 0.01 \cdot i\Big), \quad i = 1, 2, \ldots, 100 \tag{15.1}$$

Other sets of hypothesis are possible as well. For example, we could pose the possible answers:

H_1 :	$p = 0.5$	(i.e. coin is fair)
H_2 :	$p \neq 0.5$	(i.e. coin is unfair)

In general, hypotheses about continuous parameters should always take the form of the parameter being in some interval or range. Any hypothesis about an unknown continuous parameter having a *specific* value, i.e. a single real number, can't be true. As we will see in Ch. 22, the common usage of NHST in fact assumes a specific value for continuous parameters, which is actually a logical flaw.

This set of hypotheses is quite poor, however, because H_1 cannot be true, since it is impossible that $p = 0.500000000\ldots$ exactly. We already know H_2 is correct without even examining the coin, and hence framing our possible answers in this way doesn't lead to any useful scientific results.

So which set of hypotheses should we use? It just depends on the required accuracy in determining p, and also on the data available. We will return to this issue, but for the time being we'll use the set of hypotheses from Eq. (15.1). Next, how do we decide which of the H_i are most plausible? Before we compute the probabilities of each H_i, we need to look at some data.

15.2 Organizing Data

The type of experiment and equipment one uses depends on the thing you're trying to find out. For mass, use a scale. For temperature, use a thermometer. In investigating p, flip the coin a bunch and look at the results. However, note that p for a coin is a joint property of the coin and the precise manner in which it is flipped.

Imagine that you conducted a first experiment where you flipped the coin 12 times, and obtained the sequence $HTTHTTTHTTTT$. You then conducted a second experiment where you flipped the coin 8 times, and obtained the sequence $THTHHTHT$. The data D in this case would be the full sequence for experiment #1, and the full sequence for experiment #2. You could also think of the data D as being the aggregate results from the 20 overall flips, i.e. a sequence of 20 data points (each H or T).

Because it is sufficient to give the number of heads k in lieu of the full sequence of flips, k is known as a *sufficient statistic* of the binomial distribution. There are other cases where this applies. For example, it turns out that the sample mean and standard deviation are sufficient statistics for the normal distribution. However, not every type of distribution or situation has a sufficient statistic. Sometimes you simply need to specify *all* of the data in its entirety.

For the question "what is p?" we may not actually need all the individual flip results. For example, if the coin flips are described by the binomial distribution of Sec. 13.1, it turns out that the only data we really need are the number of heads from each experiment. Thus, *if the binomial distribution applies* it would be sufficient here to consider the data D as

Experiment 1 :	$n_1 = 12$ flips,	$k_1 = 3$ heads
Experiment 2 :	$n_2 = 8$ flips,	$k_2 = 4$ heads

In general, we may imagine N experiments, where there are n_i flips and k_i heads in experiment #i. The data D would then be the n_i and k_i for each of the experiments $i = 1, 2, ..., N$.

Why Can't We Just Compute The Value of p?

Notice that we have flipped the coin a total of $n = n_1 + n_2 = 20$ times and got a total of $k = k_1 + k_2 = 7$ heads. Since 7 of 20 flips were heads, can't we just say $p \approx 7/20 = 0.35$, or rather that p is near 0.35? Well, in the absence of prior information, it turns out that the *most likely interval* containing p is near 0.35. However, there are of course other possible intervals. The center (here 0.35) of the most-likely interval is known as a `point estimate`.

In the presence of prior information, $p = 0.35$ won't even be the center of the most likely interval. Furthermore, instead of simply computing the point estimate $p = 0.35$, we want to make statements like *"There is a 90% probability that the true value of p is between 0.18 and 0.51."* Addressing these issues requires a full Bayesian calculation.

15.3 Pick Statistical Models

What Is A Statistical Model?

A Bayesian analysis would compute the posterior probability of some H_i given the data D. For example, say we were interested in H_{35} : $p \in (0.34, 0.35)$. Bayes theorem' Eq. (11.3) gives

$$\mathbb{P}(H_{35}|D) = \frac{\mathbb{P}(D|H_{35})}{\mathbb{P}(D)} \cdot \mathbb{P}(H_{35}) = \frac{\mathbb{P}(D|H_{35})}{\sum_{i=1}^{100} \mathbb{P}(D|H_i)\,\mathbb{P}(H_i)} \cdot \mathbb{P}(H_{35})$$

As discussed in Ch. 11, there are two types of quantities we need in order to compute the posterior probability:

- the *likelihood* $\mathbb{P}(D|H_i)$, which gives the probability of obtaining the data D assuming that H_i is true
- the *prior probability* of H_i, denoted $\mathbb{P}(H_i)$, which is the probability we would assign to H_i before we look at the data D

To compute numbers for these we need to choose a `statistical model` for the situation, which in general means specifying the probability distributions and densities that describe the random variables in a problem, as well as specifying any additional equations that might relate various quantities.

An example of an additional equation might be a regression equation. For example, we might assume a quantity is modeled by a normal distribution, but we might additionally specify that the mean of this normal distribution varies linearly with some predictor quantity. We'll discuss this further in the last part of the text.

Statistical Model For Likelihood

Let's first look at the likelihood. If the coin flips are independent events, then they should be described by a binomial distribution. For any assumed value of p, we'll model the probability of getting k heads in n flips by Eq. (13.1):

$$\mathbb{P}(k|p) = \frac{n!}{k!(n-k)!} p^k (1-p)^{n-k} \tag{15.2}$$

Note that this is an equation giving the *probability* of obtaining k. We represent this by the mathematical statement

$$\boxed{k \sim \text{binomial}(n, p)} \tag{15.3}$$

where we use the symbol "$\sim$" rather than an equal sign "$=$" to emphasize that we don't know ahead of time what k will equal, but only how it is described statistically. Eq. (15.3) is the compact way of denoting this *statistical model* for the number of heads k.

To see how this statistical model lets us compute the likelihood, for the moment assume our data D is simply one experiment where we obtain $k = 15$ heads in $n = 37$ flips. For the hypothesis H_{35}, we have $p \in (0.34, 0.35)$, so let's use $p = 0.345$ to be definite. The likelihood then becomes $\mathbb{P}(D|H_{35}) = \mathbb{P}(k = 17|p = 0.345)$ for $n = 37$, and this is given by Eq. (15.2).

Statistical Model For Prior

The prior is supposed to somehow capture our prior knowledge. If we know absolutely nothing about the coin, then we have no reason to find any of the H_i more

plausible than the others. Since we have 100 hypotheses, corresponding to the 100 possible intervals for p, we would then assign a prior probability of $\mathbb{P}(H_i) = 1/100$ to each of these hypotheses. In other words, we would use a *uniform* distribution for the prior probabilities. Since our hypotheses are statements about p, we could represent our prior probabilities for the H_i by the following statistical model for p:

$$\boxed{p \sim \text{uniform}(0,1)} \tag{15.4}$$

where the 0 and 1 represent the bounds of possible values of p. On the other hand, if we knew something about the coin beforehand (such as results from prior experiments) we would use a different statistical model for the prior.

Summary of Question and Statistical Models

Notice that we don't specify a statistical model for the posterior probability, which is the probability or plausibility of an H_i given the data D. Remember that the posterior distribution $\mathbb{P}(H_i|D)$ indicates how the total plausibility or probability of 1 is spread out (or distributed) amongst the various hypotheses H_i. But the way in which this probability is distributed isn't something you *specify* ahead of time – it's the thing that you're *computing* via Bayes' theorem.

Question:	What is p?
Possible Answers:	$\underbrace{p \in (0.00, 0.01)}_{H_1}$ $\underbrace{p \in (0.01, 0.02)}_{H_2} \ldots \underbrace{p \in (0.99, 1.00)}_{H_{100}}$
Data:	$D = k_i$ heads in n_i flips, $i = 1, 2, \ldots, N$ experiments
Statistical Models:	$k \sim \text{binomial}(n, p)$ $\leftarrow$ likelihood $= \mathbb{P}(D\|H_i)$
	$p \sim \text{uniform}(0,1)$ $\leftarrow$ prior $= \mathbb{P}(H_i)$

15.4 Compute Posterior Probabilities

We can now use Bayes' theorem to compute the posterior probability distribution for the H_i. Computing this distribution amounts to computing the individual posterior probability for each H_i. Following the discussion in Sec. 15.3, let's for the moment focus on H_{35}. Remember this hypothesis says that p is somewhere between 0.34 and 0.35. Bayes' theorem Eq.(11.3) gives

$$\mathbb{P}(H_{35}|D) = \frac{\mathbb{P}(D|H_{35})}{\mathbb{P}(D)} \cdot \mathbb{P}(H_{35}) = \frac{\mathbb{P}(D|H_{35})}{\sum_{i=1}^{100} \mathbb{P}(D|H_i)\,\mathbb{P}(H_i)} \cdot \mathbb{P}(H_{35}) \tag{15.5}$$

Let us assume that the data D is what was specified in Sec. 15.2, so that in experiment 1 there were $k_1 = 3$ heads in $n_1 = 12$ flips, and in experiment 2 there were $k_2 = 4$ heads in $n_2 = 8$ flips.

Computing The Likelihood $\mathbb{P}(D|H_{35})$

Remember that the likelihood is the probability of getting the data D if we assume the hypothesis H_i is true. The hypothesis H_{35} says that p lies between 0.34 and 0.35. So what value would we use for p? Here we will just take an average of 0.34 and 0.35, and use $p = 0.345$.

The best answer about what to pick for p is to make the question irrelevant by choosing the intervals for p to be sufficiently small. For example, if we had chosen a hypothesis H_{350} to be $p \in (0.349, 0.350)$, then it doesn't really matter whether we use 0.349 or 0.350 in the binomial formula, since the answers will be virtually identical.

We'll denote this value of $p = 0.345$ from H_{35} as p_{35}. The probability of obtaining

$k_1 = 3$ heads in $n_1 = 12$ flips in the 1st experiment is then

$$\begin{aligned}\mathbb{P}(k_1|p_{35}) &= \frac{n_1!}{k_1!(n_1-k_1)!} \cdot p_{35}^{k_1} \cdot (1-p_{35})^{n_1-k_1} \\ &= \texttt{dbinom(3,12,0.345)} \\ &= \frac{12!}{3!9!} \cdot 0.345^3 \cdot 0.655^9 = 0.2004711\end{aligned}$$

Likewise, the probability of obtaining $k_2 = 4$ heads in $n_2 = 8$ flips in the 2nd experiment, again assuming the p from H_{35}, is

$$\begin{aligned}\mathbb{P}(k_2|p_{35}) &= \frac{n_2!}{k_2!(n_2-k_2)!} \cdot p_{35}^{k_2} \cdot (1-p_{35})^{n_2-k_2} \\ &= \texttt{dbinom(4,8,0.345)} \\ &= \frac{8!}{4!4!} \cdot 0.345^4 \cdot 0.655^4 = 0.1825323\end{aligned}$$

The probability of obtaining both of these results is given by Eq. (10.4) as

$$\mathbb{P}(k_1|p_{35}) \cdot \mathbb{P}(k_2|p_{35}),$$

in other words by the *product* of these two probabilities. Thus, the likelihood for hypothesis H_{35}, which is a measure of compatibility between the data D and this hypothesis, is given by

$$\begin{aligned}\mathbb{P}(D|H_{35}) &= \mathbb{P}(k_1|p_{35}) \cdot \mathbb{P}(k_2|p_{35}) \\ &= \texttt{dbinom(3,12,0.345)} \cdot \texttt{dbinom(4,8,0.345)} \\ &= 0.2004711 \cdot 0.1825323 = 0.03659243\end{aligned} \tag{15.6}$$

Computing The Prior $\mathbb{P}(H_{35})$

To compute the prior, we note that $p \sim \text{uniform}(0,1)$, meaning that all equal-width intervals for p are equally probable, meaning all the H_i are equally probable. Since there are 100 of the H_i, any individual hypothesis must have a probability of $1/100$. Thus, as stated before, $\mathbb{P}(H_{35}) = 1/100$.

Computing The Denominator $\mathbb{P}(H_{35})$

Now we come to the denominator in Eq. (15.5). This denominator, $\mathbb{P}(D)$, is the probability of obtaining the data D. To compute this, we need to find all the different ways we could get D, and add up probabilities. This involves summing over all the possible hypotheses that could have yielded D. Thus, we essentially need to do the computation of Eq. (15.6) for every possible H_i, and add them all up. For many problems, doing this summation is very time-consuming, and using computers is pretty much a requirement. Our denominator, remembering that k_1 is the number of heads in n_1 flips in experiment 1, etc., is

One can actually do the sum for our current problem by hand, and obtain a final simple formula for the posterior. While there are some simple problems where this works (especially if you add in some knowledge of calculus), there are many problems where you can't do it by hand, so in this book we simply adopt the approach of doing it all by computer from the start.

$$\begin{aligned}\mathbb{P}(D) &= \sum_{i=1}^{100} \mathbb{P}(D|H_i) \cdot \underbrace{\mathbb{P}(H_i)}_{=1/100} \\ &= \sum_{i=1}^{100} \mathbb{P}(k_1|p_i) \cdot \mathbb{P}(k_2|p_i) \cdot \frac{1}{100} \\ &= \sum_{i=1}^{100} \frac{\frac{n_1!}{k_1!(n_1-k_1)!} p_i^{k_1} (1-p_i)^{n_1-k_1} \cdot \frac{n_2!}{k_2!(n_2-k_2)!} p_i^{k_2} (1-p_i)^{n_2-k_2}}{100}\end{aligned} \tag{15.7}$$

We will compute this in R. To begin with, if we continue to use for each H_i the middle value of p in the corresponding interval, then we see that

$$\begin{aligned} p_1 &= \frac{0.00 + 0.01}{2} = 0.005 \\ p_2 &= \frac{0.01 + 0.02}{2} = 0.015 \\ p_3 &= \frac{0.02 + 0.03}{2} = 0.025 \\ &\vdots \\ p_{99} &= \frac{0.98 + 0.99}{2} = 0.985 \\ p_{100} &= \frac{0.99 + 1.00}{2} = 0.995 \end{aligned}$$

In other words, the p_i form a sequence that starts at 0.005 and ends at 0.995 with spacing 0.01. We can easily generate a vector of such p in R via

```
p <- seq(from=0.005,to=0.995,by=0.01)
```
(R.15.1)

With this vector in place, we can now compute the above sum via the command

```
sum( dbinom(3,12,p)*dbinom(4,8,p)*1/100 )
```
(R.15.2)

to obtain $\mathbb{P}(D) = 0.00946$.

Putting It All Together

We now put it all together, and find that

$$\begin{aligned} \mathbb{P}\left(H_{35}|D\right) &= \frac{\mathbb{P}\left(D|H_{35}\right) \cdot \mathbb{P}(H_{35})}{\mathbb{P}(D)} \\ &= \frac{0.03659243 \cdot \frac{1}{100}}{0.009459924} \\ &= 0.0387. \end{aligned} \tag{15.8}$$

We have concluded that, given the data D, the probability that p is between 0.34 and 0.35 is 3.87%. We can now repeat the same calculation for all the other H_i. This isn't as bad as it seems, because each of these calculations will involve the same denominator $\mathbb{P}(D)$, and we've already calculated it. We can actually compute a vector with all the posterior probabilities, in order from $\mathbb{P}(H_1|D)$ to $\mathbb{P}(H_{100}|D)$. Here is the complete code:

```
k1 <- 3   # heads from 1st experiment
n1 <- 12  # flips in 1st experiment
k2 <- 4
n2 <- 8
p <- seq(from=0.005,to=0.995,by=0.01)      # prob. of heads
pH <- 0.01                                 # p(H)
pD <- sum( dbinom(k1,n1,p)*dbinom(k2,n2,p)*pH )  # p(D)
likelihoods <- dbinom(k1,n1,p)*dbinom(k2,n2,p)
posterior <- likelihoods*pH/pD
```
(R.15.3)

We could get the posterior probability for H_{35} simply by accessing the 35^{th} element of `posterior` via `posterior[35]`. We can also make a plot of the posterior distribution via `plot(p,posterior)`, as shown in Fig. 15.1.

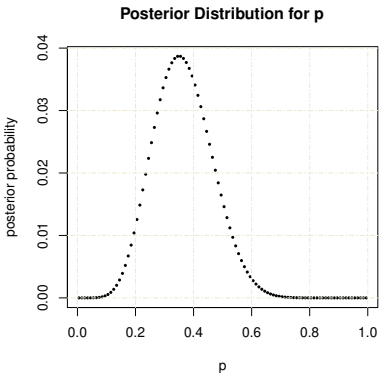


Figure 15.1: Posterior probability distribution for p. Rather than framing our hypotheses in terms of continuous values of p, we have instead chosen 100 hypotheses, each corresponding to an interval of width 0.01 of possible values of p. As such, the plot shows 100 points, one for each hypothesis, rather than a continuous curve.

15.5 Credible Intervals

Inspecting the `posterior` plotted in Fig. 15.1 shows that it is peaked at $p = 0.355$, corresponding to hypothesis $H_{36} : p \in (0.35, 0.36)$. This is consistent with the point estimate $p = 0.35$ we discussed in Sec. 15.2. Typing `posterior[35]` gives 0.039, meaning *there is a 3.9% probability the true value of p is in the interval* $0.35 < p < 0.36$. In a Bayesian analysis, an interval and corresponding posterior probability is known as a `credible interval`. Thus, $(35, 36)$ is a 3.9% credible interval for p.

It is important to understand that the *confidence interval* of frequentist statistics is *NOT* the same thing as the Bayesian credible interval. We will discuss confidence intervals in Ch 21.

We can compute other credible intervals as well. Let's say we start at H_{36}, the maximum, and go 5 hypotheses in each direction, and add up posterior probabilities. In other words, add $\mathbb{P}(H_{31}) + \mathbb{P}(H_{32}) + \cdots + \mathbb{P}(H_{40}) + \mathbb{P}(H_{41})$. We can do this in R via

(R.15.4)
```
index.lower <- 36 - 5
index.upper <- 36 + 5
sum(posterior[index.lower:index.upper])
```

to obtain 0.408. Remembering that H_{31} means $p \in (0.30, 0.31)$ etc, this means there is a probability of 40.8% that p lies in the range 0.30 to 0.41. Thus, $(0.30, 0.41)$ is a 40.8% `credible interval` for p.

What if we wanted to find something closer to an 80% credible interval? Instead of going 5 hypotheses in each direction, go some other number in each direction. Trying out different numbers, we find going 12 in each direction gives 0.78, and going 13 in each direction gives 0.82. In other words, $\sum_{i=23}^{49} \mathbb{P}(H_i) = 0.82$, as can be computed via `sum(posterior[(36-13):(36+13)])`. Thus, $(0.22, 0.49)$ is an 82% credible interval for p, meaning that there is a probability of 82% that p lies in the range $(0.22, 0.49)$.

When we transition to dealing with continous posterior densities in subsequent chapters, we'll have access to more convenient methods of computing credible intervals via quantile functions. For us, the important point here is that, in general, *it is insufficient to simply give point estimates for some quantity; we also need to provide an interval and an associated probability.* Ultimately, the purpose of the Bayesian calculation is to attach some probability or plausibility to such intervals.

15.6 Posterior Density Functions

Eq. (15.5) gives a probability $\mathbb{P}$ for $p \in (0.34, 0.35)$, i.e. for the probability that p is in a narrow window $\delta = 0.01$ near $p = 0.34$. However, we are allowed to break up the total interval $p \in (0, 1)$ into a much larger number of intervals. When we do this, we end up with a much larger number of points in the plot of the posterior distribution Fig. 15.1, and it starts to approach a continuous curve instead of a collection of separate dots. In other words, we transition from a discrete probability distribution to a continuous probability density ρ, as introduced in Ch. 14.

The relationship between our discrete model and the exact continuous model is

$$\mathbb{P}\Big(p \in (0.34, 0.35)\Big) \approx \rho\,(p = 0.34) \cdot 0.01 \tag{15.9}$$

or more generally

$$\mathbb{P}\Big(p \in (p, p+\delta)\Big) = \rho\,(p) \cdot \delta \tag{15.10}$$

where the RHS of Eq. (15.10) is really an area, where the values of ρ and δ are the height and width of a tiny rectangle. This follows the rules set forth in Ch. 14: when using a continuous probability density to find the probability of a parameter to be in some interval, find the corresponding area under the density curve.

In a similar manner, we can express the prior distribution of the H_i with a density function. We can then rewrite expressions like Eq. (15.5) as

$$\rho\,(p|D) \cdot \delta = \frac{\mathbb{P}\,(D|p) \cdot \rho\,(p) \cdot \delta}{\mathbb{P}(D)} \tag{15.11}$$

where the denominator is a sum of $\mathbb{P}\,(D|p) \cdot \rho\,(p) \cdot \delta$ over all the values of p. We see that both sides of this equation are multiplied by the same number δ, so we can just divide both sides by δ to obtain

$$\rho\,(p|D) = \frac{\mathbb{P}\,(D|p) \cdot \rho\,(p)}{\mathbb{P}(D)} \tag{15.12}$$

which is Bayes' Theorem rewritten in terms of probability density functions. Some important points here:

When computing a probability like this, the interval for p must be small enough that ρ is roughly conStant over the interval. Then the area under the curve is approximately a rectangle. To compute the probability over an interval big enough that ρ changes a lot, we must break the region into a bunch of smaller subintervals of roughly conStant ρ, and add up the areas of all these smaller rectangles.

1. The prior and posterior functions are now *probability densities* for p. Given the density ρ at some point p, you can multiply it by a small interval to get the probability that p is in that interval. For example, if $\rho = 0.3$ near $p = 0.7$, to find the probability that $p \in (0.69, 0.71)$, multiply the value of ρ by $0.71 - 0.69 = 0.02$ to get a probability of 0.006 that p is in this interval.

2. The likelihoods $\mathbb{P}(D|p)$ are still regular probabilities (as opposed to densities), because the data being modeled by the likelihood is *discrete*. This will no longer be the case when we deal with continuous data.

3. We have essentially combined *all* of the hypotheses and their corresponding probabilities into a single object – the posterior probability density.

15.7 Exercises

You are investigating the probability of heads p in a coin. You conduct 3 experiments and have the following data:

Experiment # 1:	$k_1 = 11$	$n_1 = 40$
Experiment # 2:	$k_2 = 5$	$n_1 = 20$
Experiment # 3:	$k_3 = 18$	$n_1 = 35$

Conduct a Bayesian analysis of the situation by doing the following:

1. Create a summary of question & statistical models (similar to the end of Sec. 15.3). In your analysis, use 1000 intervals instead of 100.

2. Compute a vector of posterior probabilities as in (R.15.3). Also make a plot of the posterior distribution.

3. Which of your hypotheses (i.e. intervals for p) has the highest probability? What is this probability?

4. Find the 80% (approximately) credible interval for p. Also find the 90 and 95% credible intervals. What happens to the interval as you raise the probability value? What is the significance of this for statistical inference?

CHAPTER 16

Computing Posteriors With RStan

In the previous chapter we conducted a Bayesian analysis of coin-flip data using R. Even for the simple case of discrete binomial data the calculation was somewhat complicated. In more complex situations it becomes *very* difficult to compute the sum involved in $\mathbb{P}(D)$ from Eq. (15.5). There's a wide variety of software that help with Bayesian calculations. In this book we'll use Stan, a program which interfaces with R through the package `rstan`. Visit the rstan website [9] for instructions on installing `rstan`. In this chapter we are going to reproduce the calculation of the previous chapter using `rstan`.

16.1 Specify Data In R Script

We will create an R script and specify the data. Stan works better when we use vectors for similar quantities. Thus, we will create vectors `nf` and `k` giving the number of flips and heads in each experiment:

(R.16.1)
```
ne <- 2         # number of experiments
nf <- c(12,8)   # num flips per experiment
k <- c(3,4)     # num heads per experiment
```

16.2 Write Stan File

We will now write a Stan file, which has a syntax that is different from R. First create a file named `binom.stan`, noting that the file extension is `.stan`. A basic Stan file has three sections: the data, the parameters (like p), and the statistical models. A bare-bones template Stan file looks like

(R.16.2)
```
data { }
parameters { }
model { }
```

Because Stan code uses a different syntax than R code, we will not use our Standard color syntax highlighting when listing Stan code.

where the appropriate stuff goes in-between the braces for each section. Let's actually space stuff out and start the template as

(R.16.3)
```
data {

}

parameters {

}

model {

}
```

Let's start by specifying the data, which are the `k` and `nf` vectors specified in the R file. Stan needs to know the length of these vectors ahead of time. Each vector has length 2, since there are two experiments. So the first thing we'll define is the number of experiments. Let's use the same name from the R script, and denote the number of experiments as `ne`. The proper Stan syntax is `int ne;` where `int` means integer. The semicolon at the end is necessary. Furthermore, it helps things if we tell Stan that `ne` is always a positive number (i.e. can't be less than zero), so we'll enter `int<lower=0> ne;`. The `data` section becomes

```
data {                                                    (R.16.4)
     int<lower=0> ne;
}
```

When we eventually call Stan from our R file, we'll make sure to let Stan know what the value of `ne` is. Next we define our `k` and `nf` vectors:

```
data {                                                    (R.16.5)
     int<lower=0> ne;
     int<lower=0> nf[ne];
     int<lower=0> k[ne];
}
```

The syntax of `k[ne]` means that `k` is a vector with `ne` elements. Now let's specify the parameters. In our case the only parameter is p, the probability of heads that characterizes the binomial distribution. Note that p is a real number between 0 and 1. The `parameters` section of the Stan file becomes

If you're unsure about what things are data and what things are parameters in your Stan code, remember that parameters have specific yet unknown actual values we're trying to find. The number of flips is known, so we regard `nf` as data rather than as a parameter. The number of heads `k` is also not a parameter, not only because it is known data, but also because this is a random variable that changes from experiment to experiment, and doesn't have any sort of fixed value. The probability of heads `p` *is* a parameter, because it is a fixed specific number we're trying to find.

```
parameters {                                              (R.16.6)
     real<lower=0,upper=1> p;
}
```

Finally we need to specify our statistical models. Recalling from Sec. 15.3 that we're modeling the likelihood as a binomial distribution and the prior as a uniform distribution, our final `model` section would look like

```
model {                                                   (R.16.7)
     k ~ binomial(nf,p);
     p ~ uniform(0,1);
}
```

We can save the file and head back to our R script. Here's the finished Stan file:

```
data {                                                    (R.16.8)
     int<lower=0> ne;           // num. experiments
     int<lower=0> nf[ne];       // num. flips per experiment
     int<lower=0> k[ne];        // num. heads per experiment
}

parameters {
     real<lower=0,upper=1> p;   // prob. of heads
}

model {
     k ~ binomial(nf,p);        // likelihood
     p ~ uniform(0,1);          // prior
}
```

Comments in a `.stan` file begin with two forward slashes.

16.3 Call Stan From R

To call Stan from R, we add the following commands to our R script (R.16.1):

(R.16.9)
```
library(rstan)
model.file <- "binom.stan"  # file is in default R directory
stan.results <- stan(file = model.file,
                     data = list(ne=ne,nf=nf,k=k),
                     iter = 2e4,
                     chains = 4,
                     cores = 2)
```

Here `model.file` is a variable containing the file path of the Stan file. Inside the `stan()` function we set the `data` parameter equal to the list of data we're sending to Stan. In the statement `ne=ne`, the `ne` on the left is the name that will be used by Stan, and the `ne` on the right is the name of our variable in the R script. For simplicity we kept names the same between R and Stan. Finally, `stan.results` is the variable into which we're storing all the results from Stan's computation. See margin note for more info on parameters.

A note on parameters to `stan()`:

1. Adjust `iter` to be a higher number for more accuracy. The higher you make it the longer the calculation will take.
2. `chains` is the number of independent calculations performed. These are combined at the end. The default is 4. You can make this higher if you have a lot of CPU cores. More chains tends to mean more accuracy.
3. Adjust the `cores` parameter to match the number of CPU cores on your machine. If you're not sure what this is, within R you can issue the commands `library(parallel)` and then `detectCores()` and it should give you a number.

If Stan is successfully installed, we should now be able to run our R script. Depending upon the speed of the computer being used, it may take a few minutes for Stan to complete its computations. To get a basic overview of the results, type `stan.results` and hit enter. In the middle of the output you should see something like the following:

(R.16.10)
```
       mean se_mean   sd   2.5%    25%    50%    75%  97.5%
p      0.36    0.00 0.10   0.18   0.29   0.36   0.43   0.57
```

The numbers from the table might change slightly each time you call Stan, since the calculational methods it uses are approximate, and can vary slightly. They shouldn't change enough that it matters. If they do, increase the value of `iter` until the numbers don't change from run to run. If you make `iter` too high you may run out of computer memory, but that shouldn't be an issue for the relatively simple situations discussed in this book.

These numbers are the mean, standard error of the mean (see Sec. 20.8), standard deviation, and some quantiles of the posterior probability density $\rho(p|D)$. We can see that the tails to the left of 0.18 and to the right of 0.57 contain a total of 5% of the total probability, meaning that 95% of the probability is between those numbers. Hence, we can immediately read from this info that $(0.18, 0.57)$ is a 95% credible interval for p. Likewise, $(0.29, 0.43)$ is a 50% credible interval for p.

To get quantiles other than 2.5%, 25%, 50%, 75%, and 97.5%, we first need to discuss the idea of *draws* from a distribution or density.

16.4 Draws From a Distribution or Density

Stan does not return a formula for the posterior density, but actually returns (among other things) a huge list of `draws` from this density. The idea of drawing from a distribution can be illustrated with an example: if we conduct an experiment where we flip a $p = 0.35$ coin 30 times, the number of heads k we get is statistically modeled by a binomial distribution: $k \sim \text{binomial}(30, 0.35)$. Each time we conduct such an experiment we will get some value of k that can be thought of as being *drawn* from the binomial distribution, in the same way that one can think of a card being drawn from a deck or a ball being drawn from an urn.

If we are given a huge list of draws from a probability density, we can effectively reconstruct the density. For example, we can use the command `rnorm()` to generate

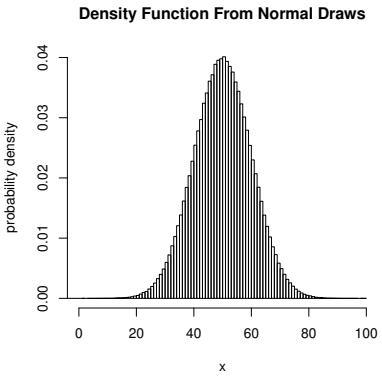


Figure 16.1: Histogram of 1e6 draws from a normal density function with $\mu = 50, \sigma = 10$. The histogram simply gives an approximation to the normal density from which the draws were taken.

simulated draws from a normal density. The command `x <- rnorm(1e6,50,10)` will generate one million simulated draws from a normal density with mean $\mu = 50$ and $\sigma = 10$, and store them to a vector `x`. If we then turn around and plot a histogram of `x` via the command `hist(x,breaks=100,freq=FALSE)`, we'll simply get back the shape of the normal density, as in Fig. 16.1.

To get the vector of draws from the posterior probability density of p, we use the `extract()` function on whatever variable we used to store the `stan()` output, specifying which parameter we want using dollar-sign notation:

```
11  p.draws <- extract(stan.results)$p
```
(R.16.11)

There are two primary things we'll do with vectors of draws: compute density functions, and compute credible intervals.

16.5 Compute and Plot Posterior Density

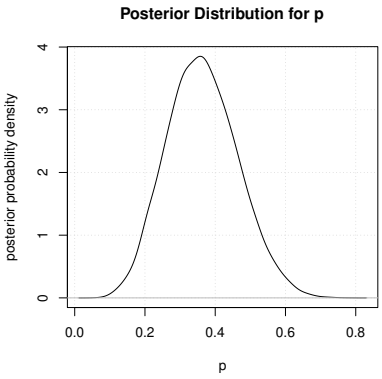


Figure 16.2: Posterior probability distribution for p. Here the smoothing parameter is set to 1.5. Because the number of iterations `iter=2e4` is relatively low, the resulting posterior will vary a bit each time you run the script. Raising `iter` will produce more stable results.

A simple way to obtain a density function from the draws is to use the function `density()`, which takes as its input a vector of draws, and returns a probability density. Thus, the command

```
12  p.dens <- density(p.draws,adjust=1.5)
```
(R.16.12)

will compute an approximate density function for the values in `p.draws`, and store it to an object we decided to name `p.dens`. The `adjust` parameter is used to smooth out the curve (higher=smoother), which may be a bit jagged due to the approximation methods employed by `stan()`. Issuing the command `p.dens` will return a quick summary of the density function.

Issuing the command

```
13  plot(p.dens)
```
(R.16.13)

will return a plot of the density function as shown in Fig. 16.2.

16.6 Computing Credible Intervals

The `quantile()` function will extract quantiles from any set of data, and doing so on the draws of p will let us calculate credible intervals. To see why this is true, note that

1. probabilities for density functions are given by areas
2. density functions are approximated by histograms of the posterior draws
3. the area of each bar in the histogram is proportional to the number of datapoints it contains (i.e. the frequency)

From these facts we conclude that the number of draws in some interval is proportional to the probability that p is in that interval. Thus, to find the 30% quantile

of the posterior distribution for p, we

1. count the total number of draws in `p.draws` and call it N
2. put all the numbers in `p.draws` in ascending order
3. start at the left and start counting until you count $0.3N$ values, and the value of p you land on is the 30% quantile

We can therefore use the `quantile()` function to compute credible intervals. For example, to compute the 64% CI for p we issue the commands

(R.16.14)
```
14 CL <- 0.64   # width of credible interval
15 pL <- 0.5 - CL/2  # left probability of interval
16 pR <- 0.5 + CL/2  # right probability of interval
17 CI <- quantile(p.draws,c(pL,pR))
```

and find that $CI_{0.64} = (0.27, 0.46)$ for p. Thus, after accounting for the data, there is a 64% probability that p is between 0.27 and 0.46.

To make sense of lines 15 and 16, consider our specific example of a 64% credible interval. To obtain an interval containing 64% of the probability, we start at the 50% mark and go 64/2=32% in each direction. The resulting quantiles of $50 - 32 = 18\%$ and $50 + 32 = 82\%$ will then contain 64% of the total probability, as you can easily check by subtracting 18 from 82.

16.7 Point Estimates

When summarizing a Bayesian analysis, in addition to giving credible intervals, it is common for people to give *point estimates*. A point estimate involves providing a single number that in some manner represents an optimal best guess for the true value of some parameter.

The sensible options are to take either the mean, median, or mode of the posterior draws / density. The mode of the posterior density is the location of the peak, i.e. the location of highest probability density ρ. The p at this location is known as the maximum a posteriori value, or MAP value for short.

In most analyses the best point estimate is given by either the mean $\bar{p}$ or the median $p_{0.5}$ of the posterior density $\rho(p)$. These can be computed via

(R.16.15)
```
18 PE.mean <- mean(p.draws)
19 PE.median <- median(p.draws)
```

which yields $\bar{p} = 0.364$ and $p_{0.5} = 0.359$. By comparison, in the margin note we compute that the MAP estimate is only $p = 0.349$. Thus, the mean and median point estimates are to the right of the peak (MAP) value for our posterior density $\rho(p)$. These three point estimates are shown in Fig. 16.3. Note that when the posterior density is perfectly symmetric, the mean, median, and mode (MAP) point estimates are equal.

The mean and median point estimates $\bar{p}$ and $p_{0.5}$ are preferred because they minimize the expected values of $(p - p_{\text{true}})^2$ and $|p - p_{\text{true}}|$, respectively, which minimizes the overall error in many types of calculations. The MAP estimate doesn't have this property, and hence isn't used as often.

To compute the MAP value, note that `p.dens` contains vectors of x and y values. Using the `which.max()` function, you can find the vector index of the maximum value of y (i.e. ρ) via `MAP.index <- which.max(p.dens$y)`, and then find the corresponding x (i.e. p) value via `p.dens$x[MAP.index]`, which yields a MAP estimate of $p \approx 0.349$.

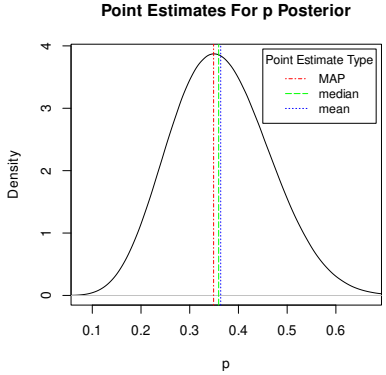


Figure 16.3: The mean, median, and mode (MAP) point estimates for p, shown against the posterior density $\rho(p)$. To get more precise results, the `iter` parameter was increased to `2e6`.

16.8 The Cumulative Posterior Density

Recall that probabilities are given by areas beneath density curves. When dealing with a probability density that's built into R, we compute areas with cumulative probability distribution functions such as `pnorm()` and `pbinom()`. We can

find the corresponding cumulative function for our situation using the R function `ecdf()`, which stands for *empirical cumulative distribution function*:

In analogy with the R practice of naming the cumulative function by putting a letter `p` in front of the name of the density, since our density is the posterior density, we are choosing to name the function `p.post`.

```
20  p.post <- ecdf(p.draws)
```
(R.16.16)

Here `p.post` is a function. For example, since we know the posterior median $p_{0.5}$ is 0.359, we expect `p.post(0.359)` to return 50%, which you can verify. We can use this function to compute probabilities in the same manner as was done in Ch. 14.

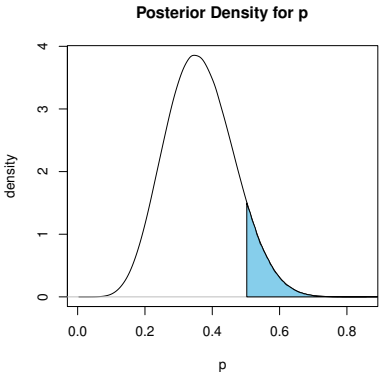


Figure 16.4: Posterior probability density function for p with area $p > 0.5$ shaded. This area is $\mathbb{P}(p > 0.5)$.

Example 16.1: Computing Probability For Posterior

Problem: For the posterior density $\rho(p)$ computed in (R.16.13), what is the probability that p is greater than 50%, i.e. $\mathbb{P}(p > 0.5)$?

Solution: $\mathbb{P}(p > 0.5)$ equals the area to the right of 0.5 on the plot of the posterior probability density for p (or equivalently one minus the areas to the left of 0.5), as shown in Fig. 16.4. Using the cumulative probability function `p.post()` computed in (R.16.16), we have

$$\begin{aligned}
\mathbb{P}(p > 0.5) &= \Big(\text{area to the right of } p = 0.5\Big) \\
&= \quad \text{[shaded]} \\
&= \quad \text{[shaded]} \quad - \quad \text{[shaded]} \\
&= 1 - \Big(\text{area to the left of } 0.5\Big) \\
&= 1 - \texttt{p.post(0.5)} \\
&= 1 - 0.905 \\
&\approx \boxed{0.095}
\end{aligned}$$

There is thus a 9.5% probability that the coin is weighted $p > 0.5$.

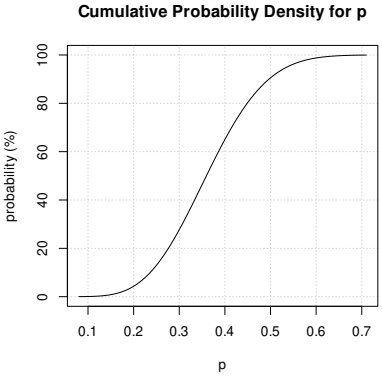


Figure 16.5: Cumulative posterior density for p.

Example 16.2: Plot Cumulative Posterior Distribution

Problem: Make a fully labeled plot of the cumulative posterior probability computed in (R.16.16). Plot the cumulative probabilities in percent.

Solution: While the `plot()` function is handy in many situations, for plotting a function, the `curve()` function is often a bit more convenient. We can set the `expr` parameter to any expression involving `x`. The code

```
curve(expr=100*p.post(x),      # multiply by 100 to plot %
      xlim=c(0.08,0.71),
      main="Cumulative Probability For p",
      xlab="p",
      ylab="probability (%)",
      panel.first=grid())
```

yields Fig. 16.5. In order to make the graph easier to read, the limits on the x-axis were chosen (after some trial and error) so as to eliminate the parts of the curve where the probability was either 0 or 1.

This graph makes it easy to directly read off various probabilities. For example, $\mathbb{P}(p < 0.4)$ is a bit above 60%, meaning there is around a 60% probability that the coin is weighted $p = \mathbb{P}(H) < 0.4$, etc.

16.9 Informative Priors

Let's build upon our calculation by incorporating some new data. In other words, we'll be doing some Bayesian updating. The procedure for computing the posterior will be exactly the same, except now we don't want to use the uniform prior for p.

So let's imagine that somebody else comes along and wants to investigate the same coin we studied previously. They decide to do three new experiments, with results summarized by

$$\begin{aligned} k_1 &= 7 \qquad n_1 = 15 \\ k_2 &= 5 \qquad n_2 = 10 \\ k_3 &= 4 \qquad n_3 = 22 \end{aligned}$$

Incidentally, it would make sense when conducting the experiments to make all the n_i the same. We're not doing it our examples simply to show that having different numbers of flips n_i doesn't pose any problems in conducting the Bayesian analysis.

Thus in experiment #1 there were 7 heads in 15 flips, etc. We should no longer use the prior $p \sim \text{uniform}(0, 1)$, because now we actually know something about the coin – the result of the previous analysis. Just as in Ch. 11, the former posterior becomes the current prior.

The posterior we saw in Ch. 16 was presented as a big list of quantiles. How could we incorporate this into a Stan prior? Well, Fig. 16.2 looks something like a bell-curve, i.e. a normal distribution. It may not actually be a normal curve, but if it's in the ballpark it's usually good enough for our prior. What would the mean μ and standard deviation σ of this normal distribution be?

The actual shape of the density curve in Fig. 16.2, which results from a binomial likelihood with a uniform prior, is what's known as a *beta-density*. If extra accuracy is needed, one should use this more exact density function, but for our purposes here the approximate normal density is good enough.

Typing `stan.results` gives the output shown in (R.16.10), indicating that the posterior has $\mu = 0.36$ and $\sigma = 0.10$. We can actually plot our posterior density together with the normal density and visually confirm that it's a pretty good match:

(R.16.17)
```
21 plot(p.dens,col="blue")
22 curve(dnorm(x,mean=0.36,sd=0.1),add=TRUE,col="red")
```

This overlay of plots is shown in Fig. 16.6. Each time Stan is run the posterior density will look slightly different, but the agreement with the overlayed normal density is reasonably good. A quick, easy, and fairly accurate way of characterizing our prior is thus

$$\boxed{p \sim \text{normal}(\mu = 0.36, \sigma = 0.10)} \tag{16.1}$$

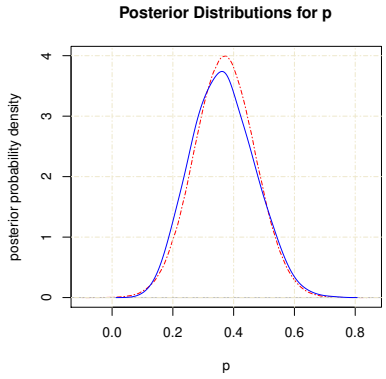


Figure 16.6: Posterior probability density for p. Here the smoothing parameter is set to 1.5. The red dotted curve is the normal density with $\mu = 0.36$ and $\sigma = 0.10$, and the blue solid curve is the posterior computed via rstan.

We will create another `.stan` file named `binom_update.stan` with the following contents:

```
data {
     int<lower=0> ne;
     int<lower=0> nf[ne];
     int<lower=0> k[ne];
}

parameters {
     real p;
}

model {
     k ~ binomial(nf,p);
     p ~ normal(0.37,0.10);  // only this line has changed
}
```
(R.16.18)

We could also set `p ~ normal(mu,sigma)` by defining `real mu;` and `real sigma;` in the `data { }` section, and passing the values of $\mu = 0.37, \sigma = 0.10$ from the R script. The advantage of this is that we can re-use the same Stan file even as the values of μ and σ change.

Note that the only change from (R.16.8) is the change in the prior for p. However, we should use new R variable names. The additional code in our script is:

```
#### perform Bayesian updating with new data ####
ne.new <- 3              # number of experiments
nf.new <- c(15,10,22)    # num flips per experiment
k.new <- c(7,5,4)        # num heads per experiment
model.file.new <- "binom_update.stan"
stan.results.new <- stan(file = model.file.new,
            data = list(ne=ne.new,nf=nf.new,k=k.new),
            iter = 2e4,
            chains = 4,
            cores = 2)
```
(R.16.19)

When specifying the list for the `data` parameter, remember that the `.stan` file variable name goes on the left of the equal sign, and the R variable name goes on the right.

We can examine quantiles of the posterior distribution for p exactly as before. The 95% credible interval for p is now $(0.24, 0.47)$. Note that this interval is *narrower* than the 95% interval of $(0.18, 0.57)$ we had before. At the 95% level we have pinned down the true value of p to a narrower range. Graphically this means that the width of the posterior density is smaller. In Fig.16.7 we plot the two posterior density: the first one that came from our first set of data using a uniform prior for p, and the new one using the new data and the updated prior. We can see that the posterior density has narrowed.

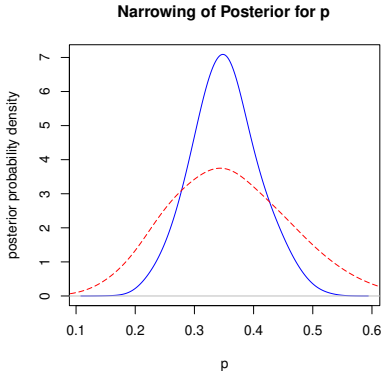


Figure 16.7: Posterior probability densities for p. Here the smoothing parameter is set to 2. The red dotted curve is the original posterior using a uniform prior for p, and the blue solid curve is the updated posterior using a normal prior for p, with μ and σ taken from the previous posterior.

The Bayesian updating procedure will often result in increasingly narrower credible intervals, meaning we're getting a better idea as to the true value of p. If new data D keeps being added into the analysis, so that the posterior for p is continually updated, we will eventually get to a very narrow interval, say a 95% credible interval like $(0.327, 0.328)$, meaning that we are getting pretty certain that the coin has p very close to 0.327 or so.

An important goal of science is to gather more and more data as we become increasingly certain about fairly narrow hypotheses, so that we might say our scientific knowledge is becoming more "concrete" or "definite". One useful feature of Bayesian methods is that it is very easy to incorporate the results of previous experiments into the analysis: just use the previous experiments to inform the prior.

16.10 Exercises

You are trying to determine the disease rate p of a particular species of tree. You have no previous knowledge of p, so as far as you know it's just as likely to be anything from 0 to 100%.

To get some information, you go to three different forests, A,B, and C, and look at 100 random[1] trees in each. In forest A you find that 32 trees are infected. In forest B you find that 37 trees are infected. In forest C you find that 29 trees are infected.

1. Clearly formulate the scientific question being asked, and the statistical models being employed. Be sure to indicate how you're separating information into *priors* and *data*.
2. Use the `rstan` package to compute the posterior density for p.
3. Make a plot of the posterior density for p.
4. Compute a point estimate for p.
5. Compute the 80%, 90%, 95%, and 99% credible intervals for p.
6. Compute a cumulative posterior probability function for p and use it to compute $\mathbb{P}(p > 0.4)$ and $\mathbb{P}(0.2 < p < 0.5)$.

Now imagine that somebody comes along with some additional data. You will perform a new analysis using the new data. In this new analysis your prior is no longer uniform, since you have prior information due to your previous experiment. The posterior density you computed in questions 2) and 3) above will serve as the prior density in your new analysis, i.e. in the following questions you will do Bayesian updating. The new data is

$$D = \begin{cases} k_1 = 35 & n_1 = 150 \\ k_2 = 53 & n_2 = 170 \end{cases}$$

7. Model the posterior $\mathbb{P}_{\text{old}}(p|D_{\text{old}})$ from questions 2) and 3) above as a normal density. What are its μ and σ?
8. Create a plot that has both the actual posterior computed by Stan as well as the normal approximation considered here. Do they seem to line up well?
9. Add code to your R script to compute the new posterior $\mathbb{P}_{\text{new}}(p|D_{\text{new}})$ using the normal prior from question 7) above. You will need an additional `.stan` file, since the prior is different. Compute the updated 80, 90, 95, and 99% credible intervals for p. How do these updated intervals compare with the previous ones from question 5?

   Since you are doing two different Bayesian analyses in a single R script, make sure to not duplicate variable names in ways that might confuse R. If you want to be extra safe, just pick totally new variable names for everything in the 2nd analysis.

10. Generate a single plot showing the original posterior density and the updated posterior density for p. How do they compare?

[1] Since disease transmission might result in clusters of infected trees, to try and have a "random sample" you ensure that the trees you look at are pretty far apart.

16.11 Appendix - Embedding Stan Model

In this Chapter we showed how to write a Stan model such as (R.16.8) in a separate file, and then use this file when calling `stan()` by using the `file` parameter. Alternatively, you can *embed* the Stan model inside your R-script by storing the model as a string, and referencing this when calling `stan()` by using the `model_code` parameter:

(R.16.20)

```
1  ne <- 2         # number of experiments
2  nf <- c(12,8)   # num flips per experiment
3  k <- c(3,4)     # num heads per experiment
4  # specify Stan model file as a string
5  model.string = "
6  data {
7       int<lower=0> ne;           // num. experiments
8       int<lower=0> nf[ne];       // num. flips per experiment
9       int<lower=0> k[ne];        // num. heads per experiment
10 }
11
12 parameters {
13      real<lower=0,upper=1> p;   // prob. of heads
14 }
15
16 model {
17      k ~ binomial(nf,p);        // likelihood
18      p ~ uniform(0,1);          // prior
19 }
20 "
21 # now call stan()
22 library(rstan)
23 stan.results <- stan(model_code = model.string,
24                      data = list(ne=ne,nf=nf,k=k),
25                      iter = 2e4,
26                      chains = 4,
27                      cores = 2)
```

Note that the Stan model specification is exactly the same as (R.16.8). We just have to enclose the whole thing with double-quotes on lines 5 and 20 in order to make it a string, and then store this to a variable name of our choice. Here we chose the name `model.string`. Whatever you call it, that's the name you use for the `model_code` parameter in line 23.

The results in `stan.results` will be the same as before. The advantage of embedding the Stan model is that it's easier to make sure you keep the R script and the associated Stan model together. The downside is that sometimes it is slower this way.

CHAPTER 17

Bayesian Group Comparisons

A common task in statistics is to see if there are differences between groups of things. For example, which species of tree is healthier, species A or species B? In this chapter, which is really a large example problem, we address this question using Bayesian inference.

17.1 Comparing Groups of Binomial Data

Imagine we are interested in planting some trees on campus, and we have to choose between species A and species B. Perhaps we are interested in which species is more susceptible to disease. Imagine that for species A, testing has been done in seven different forests or regions, with the following data:

$$n_A = 76, 45, 113, 120, 20, 95, 28$$
$$k_A = 24, 14, 45, 49, 6, 42, 10$$

meaning in the first forest, 24 of the 76 trees tested were infected, and in the next forest 14 of the 45 trees tested were infected, etc. Additionally, four forests were tested for species B, with the following results:

$$n_B = 145, 110, 45, 83$$
$$k_B = 48, 32, 19, 29$$

Based on this info, which species is healthier?

There are actually two different kinds of questions we can ask in this situation. There is a *decision*-based question we can ask: "which species should we plant?" The answer here will be a simple "A" or "B", since we must ultimately choose one of them. There is also a *scientific* question "which species is healthier?". Due to noise and other uncertainty, the answer here will be probability statement. This is discussed more in Sec. 17.5.

In answering this question, a good first step would be finding point estimates of the disease rates p for each species. In species A, of the total 497 trees tested there are 190 infected trees, giving a disease-rate point estimate of $p_A = 190/497 = 0.38$. For species B this point estimate is $p_B = 128/383 = 0.33$. Species B has the lower point-estimate disease rate p_B, and hence would appear to be the healthier species.

However, we know that point estimates aren't the full story, among other things due to noise in the data. For example, it is possible that in reality $p_A < p_B$, but we just happened to look at samples of trees for which $p_A > p_B$. We will need probability calculations to quantify our uncertainty.

17.2 Compute Posteriors for A and B

The R script for entering the data and calling Stan is

```
# input data
nFA <- 7     # number of forests for species A
nA <- c(76,45,113,120,20,95,28)  # num. trees tested
kA <- c(24,14,45,49,6,42,10)     # num. trees w/ disease
nFB <- 4
nB <- c(145,110,45,83)
kB <- c(48,32,19,29)
# call stan()
library(rstan)
stan.results.A <- stan(file = "trees-compare.stan",
                       data = list(nF=nFA,n=nA,k=kA),
                       iter = 2e4, chains = 4, cores = 2)
stan.results.B <- stan(file = "trees-compare.stan",
                       data = list(nF=nFB,n=nB,k=kB),
                       iter = 2e4, chains = 4, cores = 2)
```
(R.17.1)

Remember that when specifying the `data` paramter, the variable name you specify in the Stan file goes on the left, e.g. `nF`, and the R object you're sending to Stan goes on the right, e.g. `nFA`.

As usual, adjust `cores` to match your computer. If you have a 4-core machine, specifying 4 instead of 2 will speed things up.

and the corresponding Stan script, named `trees-compare.stan`, is

```
data {
  int<lower=0> nF;        //# of forests
  int<lower=0> n[nF];     //# sampled trees per forest
  int<lower=0> k[nF];     //# infected trees per forest
}
parameters {
  real<lower=0,upper=1> p;  //disease rate
}
model {
  k ~ binomial(n,p);       // likelihood
  p ~ uniform(0,1);        // prior
}
```
(R.17.2)

Notice that while the *data* is different for each forest, the *statistical model* is the same. This lets us use the same Stan model file for each, with the appropriate data passed to the model through `stan()`. Of course we are assuming that the two forests and their associated statistical models are independent of each other. If the statistical models for each were intertwined somehow, we would need to reflect this in a more complicated Stan model file that incorporates both datasets together.

We can get a quick visual of the resulting posterior densities by extracting draws and then plotting density curves, as we did in Sec. 16.5. To facilitate comparison we will plot both density curves on the same graph.

```
# extract draws from posterior density
pA.draws <- extract(stan.results.A)$p
pB.draws <- extract(stan.results.B)$p
# compute density functions
pA.dens <- density(pA.draws,adjust=1.5)
pB.dens <- density(pB.draws,adjust=1.5)
# plot density functions on same graph
plot(pA.dens,
     col = "blue",
     main = "Posterior Densities for pA, pB",
     xlab = "Disease Rate p",
     panel.first = grid())
lines(pB.dens,col="red",lty="dashed")
legend(x=0.43, y=13,
       legend=c("pA","pB"),
       col=c("blue","red"),
       lty=c("solid","dashed"))
```
(R.17.3)

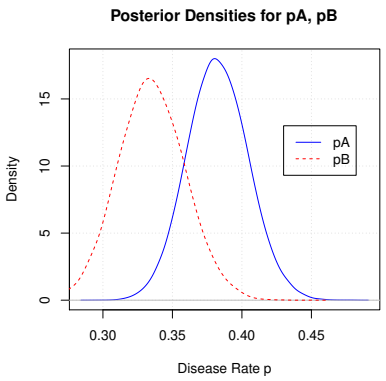


Figure 17.1: Plots of posterior densities for p_A and p_B.

We use the `lines()` function in order to add the line for `pB.dens` to the plot we already made for `pA.dens`.

The result is shown in Fig. 17.1. As can be seen by typing `stan.results.A` and

`stan.results.B` at the console, the posterior for p_B is peaked at around 0.34 with a standard deviation of around 0.02, whereas p_A is peaked around 0.38. The graph shows that the posterior density for p_A is largely to the right of p_B, suggesting that the disease rate for A is higher than that for B. We can visualize this in another way, by making box plots for posterior draws:

(R.17.4)
```
33 boxplot(list(pA.draws,pB.draws),
34         names=c("pA","pB"),
35         main="Boxplot of pA, pB Draws")
```

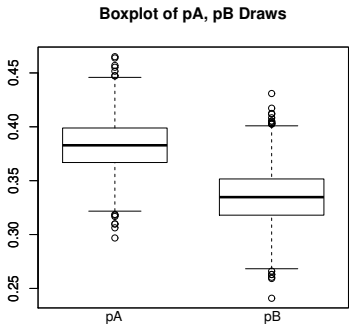


Figure 17.2: Boxplots of the draws from the posterior distributions of p_A and p_B.

with result shown in Fig. 17.2. It is clear from the boxplots that the 50% credible intervals (i.e. the interquartile regions) don't overlap. The evidence is mounting that B is healthier.

A nice way of combining the information from Fig. 17.1 and Fig. 17.2 is the `violin plot`. This type of plot shows a boxplot with the probability density overlayed. To do this in R, at the console (not in the script) install the `sm` and `vioplot` libraries with the commands `install.packages("sm")` and `install.packages("vioplot")`. In the script we then add:

(R.17.5)
```
36 library(vioplot)
37 plot(x=NA, y=NA,
38      xlim=c(0,2.05),
39      ylim=range(c(pA.draws,pB.draws)),
40      xlab="", ylab="",
41      xaxt="n",
42      main="Violin Plots of pA, pB Posterior Densities")
43 grid(nx=NA,ny=NULL)
44 vioplot(pA.draws, at=0.5, add=TRUE, col="magenta")
45 vioplot(pB.draws, at=1.55, add=TRUE, col="cyan")
46 axis(side=1, at=c(0.5,1.55), labels=c("pA","pB"))
```

Notes on the plot code lines:

37: Using `x=NA` etc. creates an empty plot window.

38: The limits on x are fairly arbitrary, and we'll pick `at` values in lines 44, 45 that are appropriate to our choice.

39: `range()` grabs the min and max values of the draws, so we can adjust the y-axis accordingly.

41: Setting `xaxt` to `"n"` suppresses numerical tick marks on the x-axis.

43: `NA` means no grid lines, and `NULL` gives grid lines at the default tick mark locations.

46: The `axis` parameter allows customization of any of the four sides of the plot. `side=1` refers to the bottom axis.

The resulting plot is shown in Fig. 17.3. Note that there is a box plot shown for p_A and p_B. Furthermore, the corresponding density curves have been turned sideways and placed on both sides of the box plot. This picture makes it clear that p_A is likely to be bigger than p_B. We can make the comparison more concrete and quantiative by computing credible intervals.

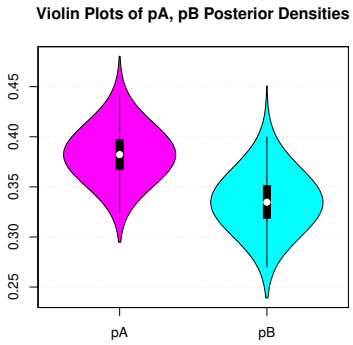


Figure 17.3: Violin plots of the draws from the posterior densities of p_A and p_B.

A Comment About Running R Code

For most of the tasks we've done until now, whenever we make changes to an R-script we simply re-run the entire script, even though only a few of the later commands may have changed. Because modern computers are so quick, this wasn't a big deal. However, `stan()` takes a while to run. If we ramp up `iter`, deal with more complicated models, or pass a large dataset to `stan()`, it could take minutes or even hours.

As such, it is best not to unnecessarily re-run `stan()` when adding to our script. In a visual environment like RStudio, one can simply select the new portions of the script with the cursor and click "Run" (or something similar) to run only the highlighted commands. This will save you a lot of time moving ahead.

17.3 Compute Credible Intervals

We can find the 95% credible intervals for p_A and p_B by adding the following to the R script:

Note that `pL` and `pR` are the probabilities corresponding to the left and right quantiles of the quantity p. It's just a coincidence the letters are the same. For example, if the posterior density were for a variable x, then `pL` and `pR` would be the probabilities corresponding to the left and right quantiles of the quantity x.

```
47  CL <- 0.95
48  pL <- 0.5 - CL/2
49  pR <- 0.5 + CL/2
50  pA.CI <- quantile(pA.draws,probs=c(pL,pR))
51  pB.CI <- quantile(pB.draws,probs=c(pL,pR))
```

(R.17.6)

Running the R-script yields the results

$$p_A: \quad CI_{0.95} = (0.34, 0.43)$$
$$p_B: \quad CI_{0.95} = (0.29, 0.38)$$

We could of course compute other credible intervals as well – there is nothing special about the number 95. In essence, the posterior density curves can be thought of as containing the information needed for computing *all* possible credible intervals. Hence, when presenting your statistical analysis it is often a good idea to present both a plot of the posterior density and also some credible intervals.

Thus, after accounting for the data, there is a 95% probability p_A is between 0.34 and 0.43, etc. There seems to be a decent amount of spread at the 95% credible level, and the intervals have a good deal of overlap. The comparison becomes clearer if we consider the posterior density for the difference between p_A and p_B.

17.4 The Posterior for `dp = pA - pB`

Of course we could also define $dp \equiv p_B - p_A$. If we did, it just means that p_A being larger than p_B gives $dp < 0$ instead of $dp > 0$. Our final results, i.e. things like the posterior probability $\mathbb{P}(p_A > p_B)$, won't depend upon which definition of dp we choose.

We can get the clearest quantitative picture of the situation if we deal directly with the quantity

$$dp \equiv p_A - p_B$$

representing the difference in the disease rates. If $dp \approx 0$ then the two species are equally healthy. If $dp > 0$ then there is a higher disease rate for A, etc.

This method for obtaining posterior draws for `dp` will NOT work if we `sort()` the vectors `pA.draws` and `pB.draws`, because then the subtraction will no longer involve "randomly" selected draws.

One way of obtaining a draw from the posterior of dp would be to take a random draw from the posterior for p_A, then take a random draw from the posterior for p_B, and subtract them. If we do this a bunch, we will end up with a bunch of draws from the posterior for dp, and from this we can get the density function, credible intervals, etc. Here we note that the draws from p_A and p_B computed via R and Stan are *already random*, i.e. not ordered in any way, and hence we could immediately obtain a good sampling of random draws from the posterior for dp via

```
52  dp.draws <- pA.draws - pB.draws
```

(R.17.7)

With `dp.draws` computed, we can examine and analyze the posterior for dp using the same methods used to analyze the posteriors for p_A and p_B separately. For example, we can make a plot of the posterior density for dp, as shown in Fig. 17.4 (the code is left to you as an exercise). A vertical line is drawn at $dp = 0$, i.e. where $p_A = p_B$. As we'll show in Ex. 17.2, most of the area (93%) is to the right of the green line in Fig. 17.4, indicating that it's very likely that $p_A > p_B$.

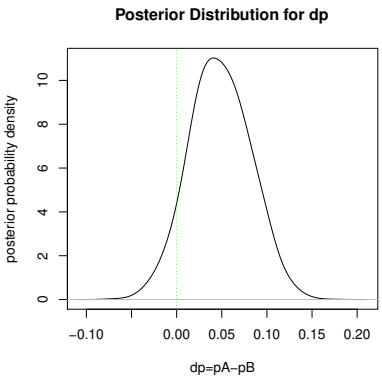


Figure 17.4: Posterior probability density for $dp \equiv p_A - p_B$. A vertical line is shown at $p_A = p_B$.

Example 17.1: Compute Credible Intervals

Problem: Compute the 95% credible interval for $dp \equiv p_A - p_B$. Also, compute the credible interval of maximum probability that contains only positive values of p.

Solution: To compute the 95% credible interval, we follow (R.16.14) and find the quantiles at the endpoints of the interval:

```
CL <- 0.95   # width of credible interval
pL <- 0.5 - CL/2  # left quantile of interval
pR <- 0.5 + CL/2  # right quantile of interval
CI <- quantile(dp.draws,c(pL,pR))
```

which yields $\boxed{CI_{0.95} = (-0.016, 0.11)}$. Adjusting `CL` (through trial and error) we find that dp is all positive at a level of 85.8%, where we have $\boxed{CI_{0.858} = (0.000, 0.095)}$. Thus, at the 85.8% credible level, dp is greater than zero, i.e. p_A is greater than p_B.

17.4.1 Cumulative Probability Function for dp

Following (R.16.16), we compute the cumulative probability function for dp via

(R.17.8)
```
53  p.post.dp <- ecdf(dp.draws)
```

The name `p.post.dp` is a bit clunky, but it has a meaning. The `p` is chosen to match R's conventional prefix in naming cumulative probability functions, such as `pbinom()` and `pnorm()`. The `post` and `dp` are to remind you that we are dealing with the cumulative probability density of the *posterior* probability density of `dp`. Feel free to choose a simpler name in your own scripts.

and use it to compute various types of probabilites.

Example 17.2: Compute $\mathbb{P}(p_A > p_B)$

Problem: For the Bayesian analysis discussed in this chapter, compute the probability that tree A has a higher disease rate than tree B.

Solution: We need to compute $\mathbb{P}(p_A > p_B)$. We first note that $p_A > p_B$ is the same as $p_A - p_B > 0$, i.e. $dp > 0$. We thus want to compute the area under the density curve to the right of zero. Using the cumulative function `p.post.dp` defined in (R.17.8), we have

$$\begin{aligned}
\mathbb{P}(dp > 0) &= \Big(\text{area to the right of } dp = 0\Big) \\
&= \quad \text{ } \\
&= \quad \text{ } \\
&= 1 - \Big(\text{area to the left of } 0\Big) \\
&= 1 - \texttt{p.post.dp(0)} \\
&= 1 - 0.07 \\
&\approx \boxed{0.93}
\end{aligned}$$

After accounting for the data, there is a 93% probability that the disease rate for A is higher than that of B, i.e. that $p_A > p_B$.

Example 17.3: Verify Credible Interval

Problem: Verify that the interval $p \in (-0.016, 0.11)$ from Ex. 17.1 contains 95% of the posterior probability for dp.

Solution: Again using `p.post.dp` from (R.17.8), we have

$$\begin{aligned}
\mathbb{P}(-0.016 \leq dp \leq 0.11) &= \text{area between } -0.016 \text{ and } 0.11 \\
&= \\
&= \Big(\text{area between } -1 \text{ and } 0.11\Big) \\
&\qquad - \Big(\text{area between } -1 \text{ and } -0.016\Big) \\
&= \quad - \\
&= \texttt{p.post.dp(0.11)} - \texttt{p.post.dp(-0.016)} \\
&= 0.973 - 0.247 \approx 0.95
\end{aligned}$$

which verifies that there is a 95% probability that $dp \equiv p_A - p_B$ is between -0.016 and 0.11.

Since $dp \equiv p_A - p_B$, the maximum value of dp is $1 - 0 = 1$, and the minimum value is $0 - 1 = -1$. This is why we are computing an area starting from -1. Note that the `p.post.dp` function takes care of this automatically.

17.4.2 Probability That `dp > x`

In Ex. 17.2 we showed that $\mathbb{P}(dp > 0) = 93\%$. This is a strong indication that p_A is larger than p_B. We should go further, however, and indicate *by how much* p_A is bigger than p_B. We can summarize this information with a plot of $\mathbb{P}(dp > x)$ vs. x.

Note that

$$\begin{aligned}
\mathbb{P}(dp > x) &= \Big(\text{area to the right of } x\Big) && (17.1)\\
&= \quad (x) && (17.2)\\
&= \quad - \quad (x) && (17.3)\\
&= 1 - \Big(\text{area to the left of } x\Big) && (17.4)\\
&= 1 - \texttt{p.post.dp(x)} && (17.5)
\end{aligned}$$

To make things easier to read, we will be plotting both axes (both dp and $\mathbb{P}$) in percent. As such, our graph will interpret x as a percent. However, `p.post.dp` expects inputs to be decimals. Accordingly, we will convert x to a decimal via $x/100$ when calling `p.post.dp`. We will then multiply the final result by 100 in order to get the probability in %. Hence, we will use `curve()` (see Ex. 16.2) to make a plot of

$$100 \times (1 - \texttt{p.post.dp(x/100)}) \tag{17.6}$$

Because it will be an important summary graph of the analysis, we will do a little extra formatting on the title and axes to make for a professional-looking graph from which we can directly read estimates of various probabilities.

Note that `x/100` is used only because our plot variable `x` represents the difference dp in disease probability, and we wish to have dp show up as a percentage on the horizontal axis. For future problems where it doesn't make sense to have the variable expressed as a percentage, we wouldn't want to include the division by 100, though we still might want to multiply by the 100 at the front, as this is just converting the posterior probability to a percentage.

(R.17.9)

```
54 # create plot title with special formatting (for subscripts)
55 title <- expression(paste("Probability of dp = ", p[A],
56                           " - ", p[B], " > x"))
57 # define locations of tick marks
58 x.ticks <- seq(from=-5, to=15, by=2.5)
59 y.ticks <- seq(from=0, to=100, by=10)
60 # create graph
61 curve(expr=100*(1-p.post.dp(x/100)),
62       xlim=c(-5,15),
63       main=title,
64       xlab="x = difference in disease rate (%)",
65       ylab="Probability (%) of dp > x",
66       xaxt="n", yaxt="n",
67       panel.first=abline(v=x.ticks, h=y.ticks,
68                          col="gray", lty="dashed"))
69 # place custom tick marks
70 axis(side=1, at=x.ticks)
71 axis(side=2, at=y.ticks)
```

The resulting graph, shown in Fig. 17.5, is a compact visual summary of our entire analysis. We can read from the graph our previous result that $\mathbb{P}(dp > 0)$ is a bit over 90%, which is strong evidence that A has a higher disease rate than B. We can also read off the probabilities of p_A being x percent higher than p_B. For example, there is around a 50% probability that disease rate p_A is more than 5% larger than p_B, but only a 20% probability that p_A is more than 7.5% larger than p_B. And so on. This graph gives a complete summary of everything we can squeeze out of the data, as regards the difference between the two species.

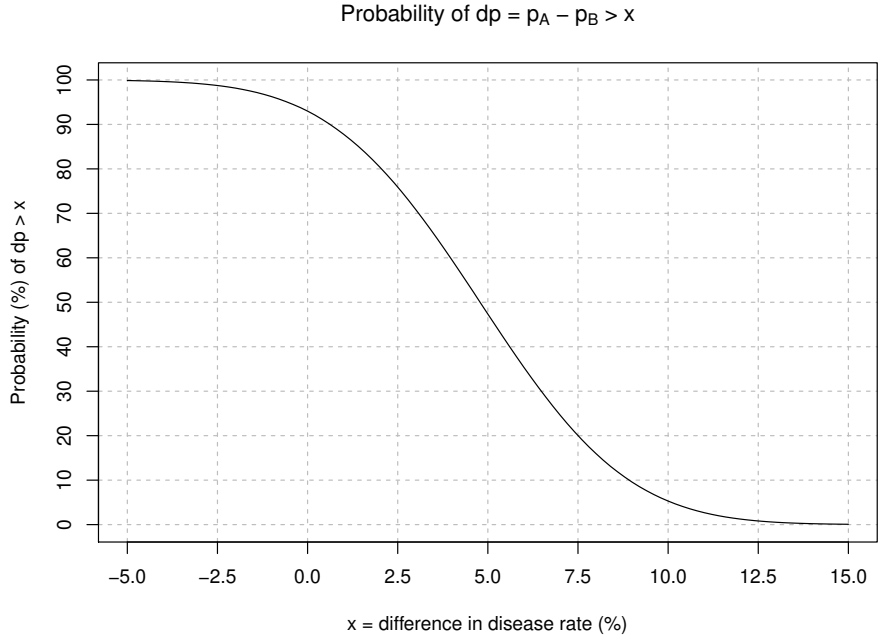


Figure 17.5: The probability that $dp \equiv p_A - p_B > x$, i.e. a plot of $1 - F(x)$, where $F(x)$ is the cumulative probability function for dp.

Notes on the plot code lines:

55,56: `paste()` combines different strings into one overall string. Inside the `expression()` function text like `p[A]` will get typeset as a `p` with a subscript `A`, i.e. like p_A. Not necessary, but it looks more professional. You can also render greek letters this way. Typing `mu` inside of an `expression()` function will typeset the letter μ, etc.

58: The *possible* values of dp range from -100% to $+100\%$. However, the posterior density is almost zero over most of this range. Inspecting Fig. 17.4 indicates that most of the posterior probability is between approx. -0.05 and 0.15. The 2.5 was chosen to give as many ticks as possible w/o being cluttered. Note that we need to define the tick locations *before* we plot with `curve()`, since we need these locations to define our grid in line 68.

66: This suppresses the default tick-marks on the axes, allowing us to add our own custom values later.

67,68: While `grid()` does a nice job of creating a default grid, when using custom tick-mark locations it is easier to use `abline()`.

70,71: The `axis()` function allows us, among other things, to add our own tick-mark locations. The `side` parameter should be set to 1 for the bottom, 2 for the left, 3 for the top, 4 for the right.

17.5 So What's Our Answer?

Our original question was

> **"Based on the data, which species is healthier?"**

At first glance this seems like a multiple-choice question, i.e. apparently the answer is either A or B. This is not the case for scientific questions. In actuality we are dealing with noisy data and other types of uncertainty. Thus, a valid answer *MUST* involve some type of probability statement that quantifies our uncertainty. Here are some valid answers to our original question:

When presenting answers, always make sure to clearly state the choice of statistical model (likelihoods, priors, etc.), since the results always depend on this choice. In our case, we should explicitly mention the binomial likelihoods and uniform priors.

> "There is a 93% probability that B is healthier than A."

or

> "The probability that A is x% more diseased than B is shown in Fig. 17.5."

or

> "The 95% credible interval for $dp \equiv p_A - p_B$ is $(-1.6, 11)$%."

On the other hand, "Which tree do we plant" is a different type of question: it isn't a scientific question, but rather indicates a *decision* to be made. When it comes to making a **decision** on which tree to plant, we clearly have to choose either A or B. But there is no math equation that can decide this for us! Ultimately it will be necessary to account for human values and judgments. For example, it seems likely that A is at most around 10% more diseased than B. Does a difference of at most 10% really matter? That's a human value judgment. Ideally you would just pick B, but what if B trees are twice as expensive? Or what if B trees don't look as nice? Etc. Ultimately people have to make a choice. The math we're doing here can't make that choice for us, but it can certainly *inform* our choice.

And these types of answers are the best we can do. If we want even more certainty that B is healthier than A, the only option is to collect more data.

If we eventually end up with an *enormous* amount of data, and many different people have repeated the experiments etc., then maybe $\mathbb{P}(dp > 0)$ becomes very close to 100%, at which point we might say that it's a *scientific fact* that $p_A > p_B$. It is not possible to attain this level of certainty in most situations, and even for so-called "scientific facts" we don't have literal 100% certainty.

17.6 Exercises

1. Write a script that creates Fig. 17.4 from the text.
2. Two coins are each flipped 50 times. Coin A comes up heads 31 times, and coin B comes up heads 25 times.
   i) Make a single plot with the posterior density for both p_A and p_B, where p_A is the heads probability for A, etc.
   ii) Give point estimates for p_A and p_B.
   iii) Make a plot of the Bayesian posterior density for $dp \equiv p_A - p_B$.
   iv) Compute the 90% credible interval for dp.
   v) Make a fully formatted plot of the density for $dp > x$ as in Fig. 17.5.
   vi) Use your graph of $dp > x$ to estimate both $\mathbb{P}(dp > 0)$ and $\mathbb{P}(dp > 15\%)$.

3. In this problem you will modify the analysis of this chapter to describe a situation with data generated by a poisson distribution instead of binomial distribution. The poisson distribution, described in Sec. 13.5, is a good description for discrete events/instances occupying finite regions of space/time.[1] An example mentioned in Ch. 13 is the number of plants existing in finite regions of space.

   A rancher is trying to determine which of two fields has a higher rate of plant growth for a specific species of plant x. They subdivide two fields, A and B, into *quadrats*. They randomly select 20 quadrats in each field, and count the number of plants k of species x in each of the selected quadrats. The count data are:

   $$A: \quad k_A = 12, 9, 3, 7, 6, 13, 7, 13, 8, 10, 8, 8, 8, 3, 9, 7, 10, 9, 9, 8$$
   $$B: \quad k_B = 9, 6, 10, 13, 8, 5, 14, 9, 9, 14, 6, 5, 7, 13, 15, 9, 9, 16, 7, 7$$

   Let λ_A and λ_B be the actual average number of plants per quadrat.

   i) Make a single plot containing the posterior densities for λ_A and λ_B. In addition to R and Stan scripts, provide a written description of the question and statistical models chosen.

      Note: When specifying your statistical models, you will need to choose prior densities for the λ_i. Unlike the binomial probability p from earlier problems, the parameter λ is not a number confined to be less than 1. You will need to therefore pick some upper limit for the λ if you choose to model the prior with a uniform density.

   ii) Compute point estimates for λ_A and λ_B

   iii) What are the 95% credible intervals for λ_A and λ_B?

   iv) Make a plot of the posterior density $d\lambda \equiv \lambda_B - \lambda_A$.

   v) Make a plot of $d\lambda > x$.

   vi) Which field has higher plant growth? Give at least 3 valid scientific answers to this question, following the discussion of Sec. 17.5.

Note that the data here is not described by a binomial distribution, but rather a poisson distribution. Hence, when defining your statistical models in the Stan file, your likelihoods should no longer involve `binomial(n,p)` but rather `poisson(lambda)`.

[1] One requirement, mirroring the requirement for the binomial distribution, is that the events/instances are *independent*. Thus, to model plant populations with a Poisson distribution, a plant existing in a region of space should be independent of whether or not other plants are nearby.

CHAPTER 18

Bayesian Inference From Normal Data

In Chapters 15 – 17 we performed Bayesian inference in situations where the data was discrete. We now turn to situations with continuous data from a normal density. In this chapter we will reexamine the crab temperature data from Sec. 1.3.

18.1 Formulating A Bayesian Analysis

We will assume that the crab temperatures T of Sec. 1.3 are statistically modeled as draws from

$$T \sim \text{normal}(\mu, \sigma)$$

where μ and σ are the mean and standard deviation of a normal density function (see Sec. 14.4). Let's first specify the question we're trying to address in our analysis:

Why are we assuming the data is drawn from a normal density? It turns out that in situations where noise arises from a bunch of *additive* pushes and pulls (see Sec. 8.1), the measurement is often described by a normal density. Thus, in the absence of any other information, it's just a reasonable guess that works out sometimes. If this assumption is wrong, hopefully our current analysis and/or subsequent data & analysis will point us in a better direction.

"What are μ and σ?"

Before we write our R script and Stan file, we pause to note this question introduces 3 new issues:

1. Unlike discrete binomial data, where the likelihood $\mathbb{P}(D|H)$ is a probability distribution, for continuous data the likelihood is a density $\rho(D|H)$. We'll just assume that Stan handles this appropriately.
2. Unlike a binomial probability p, the parameters μ and σ are not bounded between 0 and 1. In general, μ can vary from $-\infty$ to $+\infty$, and σ can vary from 0 to $+\infty$ for a normal density function. This makes Stan's job a bit tougher, and is also something we need to keep in mind when establishing priors for μ and σ.
3. The possible answers to the question specify values for both μ and σ. This is an example of a `joint posterior density`, where the "joint" means there is more than one parameter being described.

If you are curious about how continuous data is handled when using Bayes' Theorem, see Section 18.9 at the end of the chapter.

We will discuss these issues as we proceed through the analysis.

18.2 Compute Posterior Draws

Our R script should store the data and then call `stan()`:

```
T <- c(25.8, 24.6, 26.1, 22.9, 25.1, 27.3, 24.0, 24.5, 23.9,
       26.2, 24.3, 24.6, 23.3, 25.5, 28.1, 24.8, 23.5, 26.3,
       25.4, 25.5, 23.9, 27.0, 24.8, 22.9, 25.4 )
N <- length(T)
library(rstan)
stan.results <- stan(file="crab-temp.stan",
                     data=list(N=N,T=T),
                     iter=2e4,  #increase for more accuracy
                     chains=4,
                     cores=4)
```
(R.18.1)

There are other priors that could also make sense. For example, if it is already known that this type of crab has a temperature near 25° C, it could make sense to take $\mu \sim \text{normal}(25, 5)$, where the 5 was just a reasonable guess for the possible spread in μ. However, we are supposing here that we have no prior knowledge. The optimal choice of a *non-informed* prior in such cases is an open and ongoing discussion in Bayesian statistics. Check out the Wikipedia page on `Prior probability` for more info.

Before we write the Stan model file, we need to determine the priors for μ and σ. If we assume no prior info, one option would be to use uniform priors. Because μ can in principle range from $-\infty$ to ∞, we need to place some realistic limits on its value. At the very least we can assume that the mean body temperature of the crabs is above freezing and below boiling, i.e. between 0°C and 100°C. We know σ must be positive, so 0 is a safe lower limit. Because the crab cannot have a temp above boiling, it follows that σ must be a good deal smaller than 100. With these limits in place, we can now write the Stan file:

```
data {
   int<lower=0> N;        // number of data points
   real<lower=0> T[N];    // crab temperature in C
}
parameters {
   real<lower=0,upper=100> mu;     // population mean temp
   real<lower=0,upper=100> sigma;  // population sd
}
model {
   T ~ normal(mu,sigma);      // likelihood
   mu ~ uniform(0,100);       // prior
   sigma ~ uniform(0,100);
}
```
(R.18.2)

Running the R script (which calls Stan) will then compute draws from the posterior density.

18.3 Joint and Marginal Posterior Densities

We now have two vectors of draws to `extract()`:

```
mu <- extract(stan.results)$mu
sigma <- extract(stan.results)$sigma
```
(R.18.3)

To make expressions more compact we use the variable name `mu` instead of `mu.draws`, and likewise for σ. But we should remember that `mu` is not a single value for μ, but is a vector of draws from the posterior density for μ.

Before continuing with the R script, we pause to discuss the nature of the joint posterior density. As mentioned in Sec. 18.1, the Bayesian posterior density contains two parameters: μ and σ. For example, the n^{th} element of `mu` and the n^{th}

element of `sigma` should be considered as a pair of numbers drawn at the "same time" from the joint posterior density. Thus, the posterior densities for μ and σ are intertwined in a joint posterior $\rho(\mu, \sigma|D)$.

For each value (i.e. small interval) of σ, there is a corresponding posterior density for μ. Fig. 18.1 shows the conditional posterior density curves for μ, also known as $\rho(\mu|\sigma, D)$, at several different values (i.e. small intervals) of σ:

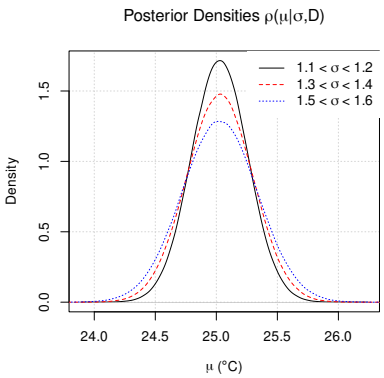


Figure 18.1: Conditional posterior densities $\rho(\mu|\sigma, D)$ at a few values of σ.

We could actually put all of these different posteriors $\rho(\mu|\sigma, D)$ together in a 3D plot, as shown in Fig. 18.2. Each of the individual $\rho(\mu|\sigma, D)$ is essentially a slice from the full joint posterior density shown in Fig.18.3.

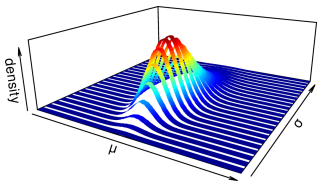


Figure 18.2: The conditional posterior densities $\rho(\mu|\sigma, D)$ assembled into a 3D graph.

Most of the time we don't care about all the different slices. Instead, we just add all of the slices $\rho(\mu|\sigma, D)$ together, i.e. for each value of μ we add up ρ over all the σ's. The result is known as the `marginal posterior density` for μ, so-called because people used to add up all the numbers in the margin of the table they had written on paper. For concision the word "marginal" is often omitted.

We return to the R script to generate plots of the marginal posterior densities:

(R.18.4)

```
# plot marginal posterior for mu
title <- expression(paste("Marginal Posterior Density For ",
                          mu))
x.title <- expression(paste(mu, " (", degree , "C) "))
plot(density(mu,adjust=1.5),
     main=title,
     xlab=x.title,
     xlim=c(23.9,26.1),
     panel.first=grid())
# plot marginal posterior for sigma
title2 <- expression(paste("Marginal Posterior Density For ",
                          sigma))
x.title2 <- expression(paste(sigma, " (", degree , "C) "))
plot(density(sigma,adjust=1.5),
     main=title2,
     xlab=x.title2,
     xlim=c(0.7,2.5),
     panel.first=grid())
```

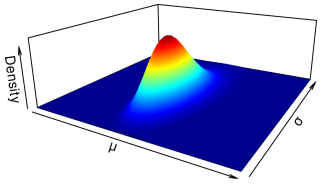


Figure 18.3: The joint posterior density $\rho(\mu, \sigma|D)$.

The resulting plots are shown in Figs. 18.4 and 18.5. Let's examine each of the posteriors in more detail.

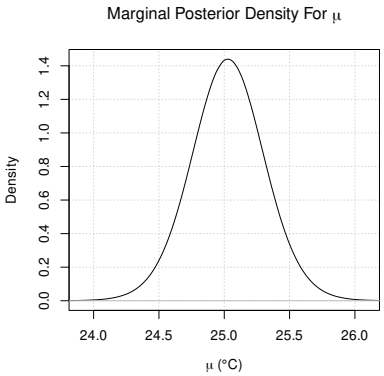


Figure 18.4: The marginal posterior density $\rho(\mu|D)$.

18.4 Posterior Density For μ

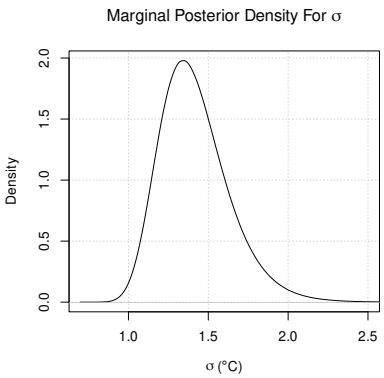


Figure 18.5: The marginal posterior density $\rho(\sigma|D)$.

While the density plot of Fig. 18.4 does a nice job of conveying the mean value of μ and also its standard deviation (which measures our degree of uncertainty about the true value of μ), it doesn't make probabilities all that apparent. Making a graph of the cumulative probability addresses this issue. As an exercise see if you can generate the plot shown in Fig.18.6. From this diagram we can, for example, estimate that the 10^{th} percentile is $\mu \approx 24.7°\text{C}$ and the 90^{th} percentile is $\mu \approx 25.4°\text{C}$, meaning there is an 80% probability that μ is between 24.7 and 25.4. Put another way, this is the 80% credible interval for μ. In this manner we can

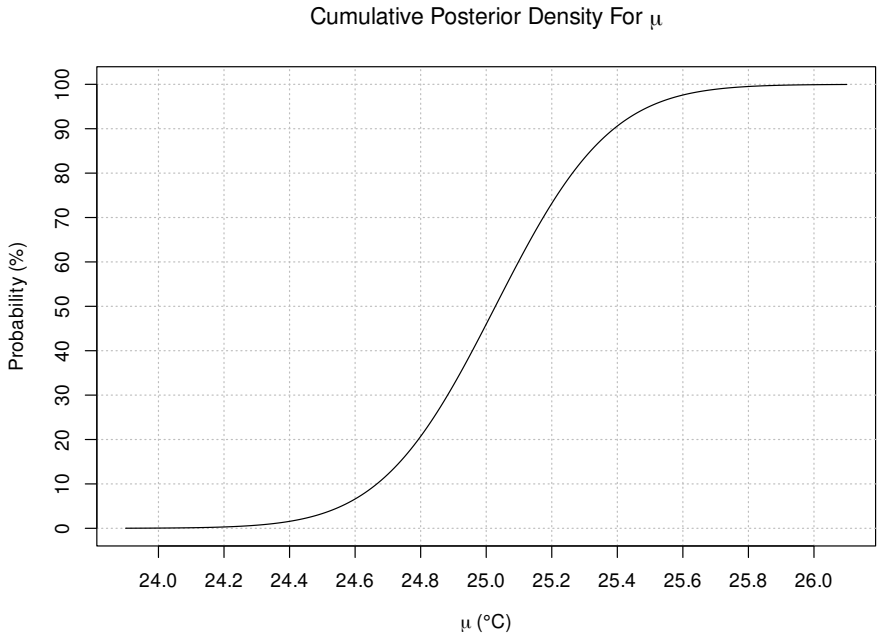


Figure 18.6: Cumulative posterior probability function for μ.

get reasonable estimates for a wide variety of credible intervals or other types of probability calculations.

We can also compute exact numbers. For example, following (R.16.14), we find that the 95% credible interval for μ is $CI_{0.95} = (24.5, 25.6)$°C. This provides one possible answer to the question of "What is μ?"

18.5 The Asymmetry of The σ Posterior

The marginal posterior density $\rho(\sigma|D)$ from Fig. 18.5 is noticeably lopsided or asymmetric around the peak. This results in different ways of defining and computing credible intervals. Here we consider two: the ETI and the HDI.

18.5.1 The ETI (Even-Tailed Interval)

All the credible intervals computed thus far involve assigning equal probability to the tails of the density. This is called an `ETI`, which stands for even-tailed interval. For example, in (R.16.14) we computed a 64% credible interval by starting at the median and going $32 = 64/2\%$ in either direction. We then find the values at the 18^{th} and 82^{nd} percentiles. Note that if there is 64% in the interval, there must be $100 - 64 = 36\%$ in the tails, and since everything is symmetric that means there is 18% in each tail. The tails are *even* in the sense that they contain the same probability.

Note that the code (R.16.14) will still compute the ETI for an asymmetric density. For the 64% example, we would would still find the values at the 18^{th} and 82^{nd}

percentiles. Because the density is asymmetric, the tails won't *look* the same or have the same width. However, since both tails contain the same area, the are still considered *even*.

18.5.2 The HDI (Highest Density Interval)

There are actually an infinite number of intervals in the posterior density that contain a given area. To generate one of them, say at the 95% level, you can arbitrarily pick a left quantile σ_L that is to the left of 5%, and then move right until you've enclosed an area of 0.95, landing on some σ_R. In general the width $\sigma_R - \sigma_L$ of these intervals will differ. The interval with the smallest width is called the `HDI`, or highest density interval. The HDI will be different from the ETI if the density curve is asymmetric.

One way to compute a HDI in R is with the code

(R.18.5)
```
library(coda)
library(HDInterval)
stan.results.mcmc <- As.mcmc.list(stan.results)
hdi(stan.results.mcmc,0.95)
```

Remember that you can use the `install.packages()` function to install a package. For example, to install the `coda` package, at the console type `install.packages("coda")` etc.

Computing the 95% ETI and HDI for μ and σ gives:

$$\text{95\% ETI:} \quad \mu \in (24.5, 25.6) \quad \sigma \in (1.07, 1.92) \tag{18.1}$$
$$\text{95\% HDI:} \quad \mu \in (24.5, 25.6) \quad \sigma \in (1.03, 1.86) \tag{18.2}$$

To within 3 signficiant digits, the ETI and the HDI for μ are the same, reflecting the fact that the posterior $\rho(\mu|D)$ is approximately symmetric. On the other hand, there is a slight difference between the ETI and HDI for σ. As is clear from Fig.18.7, the HDI is slightly narrower than the ETI, and is also shifted a tiny bit to the left. The difference between the ETI and HDI grows as the posterior becomes more asymmetric.

Even for μ the ETI and HDI aren't *exactly* the same, given that the posterior density isn't *exactly* symmetric. But in this case the differences don't show up until the 4^{th} significant figure. Since the original data is given to only 3 signficant figures, the difference between the two intervals is regarded as negligible here.

So which type of credible interval should you use when dealing with an asymmetric posterior density? The HDI is preferred by some because it results in a narrower interval, suggesting that one has "pinned down" the value of the parameter a bit better. This doesn't necessarily mean the HDI is "better" or more accurate. So feel free to use either. Just make sure to indicate which type of interval your are reporting.

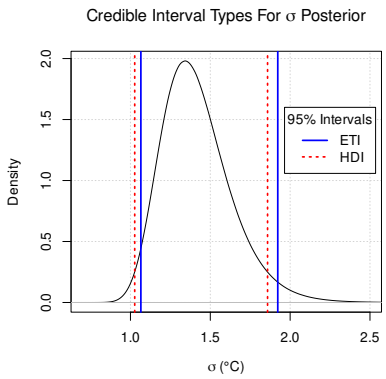


Figure 18.7: 95% ETI and HDI for σ.

18.5.3 A Comment About Point Estimates

Computing the mean and median point estimates for μ and σ gives

$$\bar{\mu} = 25.0 \qquad \mu_{0.5} = 25.0$$
$$\bar{\sigma} = 1.42 \qquad \sigma_{0.5} = 1.39$$

To a very good approximation both point estimates for μ agree, and occur at the mid-point of both the ETI and the HDI for μ. On the other hand, the two point estimates for σ differ. Furthermore, neither of these σ point estimates lies in the middle of either the ETI or the HDI. For example, the mid-point of the ETI (i.e. the mean of the end-points of the interval) is $\sigma = 1.49$ and the mid-point of the

HDI is $\sigma = 1.44$. This is all due to the fact that the marginal posterior density $\rho(\sigma|D)$ is noticeably asymmetric.

So what is the take-home lesson here? When dealing with an asymmetric posterior, do not assume a good point-estimate is given by the mid-point of either the ETI or HDI.

18.6 The Standard Deviation of Sigma?

Notice that the posterior densities for μ and σ, shown in Figs. 18.4 and 18.5, have their own means and standard deviations. We can compute these by using `mean()` and `sd()` on the draws. We end up with the following:

$$\begin{aligned} \bar{\mu} &= 25.0 \qquad & s_\mu &= 0.287 \\ \bar{\sigma} &= 1.42 \qquad & s_\sigma &= 0.220 \end{aligned}$$

where s indicates standard deviation. Sometimes it can be confusing to think of the standard deviation s of a standard deviation σ. Just remember that we are talking about two completely different things:

- The parameters μ and σ statistically model the temperatures of crabs. μ is the mean crab temperature, and σ is a measure of how much variation there would be from crab to crab.
- The quantities $\bar{\mu}$, $\bar{\sigma}$, s_μ, and s_σ refer to our *state of knowledge* regarding the values of μ and σ. Thus, while a point estimate for μ would be $\bar{\mu}$, we aren't certain this is the correct number, and our uncertainty is reflected in s_μ. Likewise, $\bar{\sigma}$ is a point estimate for σ, but we are uncertain as to the true value of σ, and this uncertainty is reflected in s_σ.

18.7 Sensitivity Analysis

Typically there are a variety of sensible choices when specifying a prior. In the current chapter we chose $\mu \sim \text{uniform}(0, 100)$. But other choices are sensible. Alternate choices might be a uniform density with different limits, or a normal density with mean near 20 and a large standard deviation (say 15).

An extremely important question in Bayesian inference is

> "How much do the results of the analysis change when we change the choice of prior?"

After all, if our results change by a lot when we making small adjustments to the prior, we have reason to suspect our analysis is unreliable. Let's try out different priors and see how the results change.

Choose A Different Uniform Prior

Suppose we changed the prior $\rho(\mu)$ to be a uniform density from 15 to 50, i.e. $\mu \sim \text{uniform}(15, 50)$. To do so, we change all instances of 0 and 100 to 15 and 50 in the Stan code (R.18.2). Re-running the R script produces results that are identical to before. The reason is simple: we are now excluding the possibility of $\mu < 15$ or $\mu > 50$. However, our original posterior density is basically zero at these values of μ.

By the same reasoning, if for some reason we set $\mu \sim \text{uniform}(15, 25)$, then our results will be radically different, as you can check. However, it's hard to imagine why somebody would use such a bizarrely narrow prior in the absense of any prior information. Generally you want to err on the side of caution and have a non-informative prior contain a wide range of possibilities.

Choose A Wide Normal Prior

We could also examine possible normal priors. In the absence of much prior information, perhaps we might guess that the crabs would have a temperature near 20°C. We are very uncertain of this, however, and hence we should choose a wide standard deviation, say 15°C. The nice thing about choosing a normal prior is that we aren't cutting off any possible values of μ. Rather, we are just assigning them low probability as they get sufficiently far from 20°C. If we compare the results from $\mu \sim \text{uniform}(0, 100)$ and $\mu \sim \text{normal}(20, 15)$, you can verify that the means of the posterior densities differ by only 0.007% (25.028 vs 25.026). So very little difference. However, point estimates are not as important as probability statements when presenting the results of a Bayesian analysis.

One way to quickly compare the probability statements that result from the two priors is to consider the difference in the cumulative probability function for each posterior density. If we let $F_1(\mu)$ and $F_2(\mu)$ represent the cumulative probabilities from the uniform$(0, 100)$ and normal$(20, 15)$ priors, then the percent difference of the normal from the uniform prior is

$$\%\text{ difference} = \left| \frac{F_2(\mu) - F_1(\mu)}{F_1(\mu)} \right| \times 100 \tag{18.3}$$

A plot of this is shown in Fig. 18.8. As an exercise see if you can use `curve()` to make a plot like this that mostly agrees apart from the left-tail (though yours will likely be bumpier, for reasons mentioned in the caption). The percent differences are only around a percent or two for most of the graph (the numbers near the left tail are less reliable). These differences are certainly larger than the 0.007% difference in the point-estimate, but are still quite small by the standards of most analyses. We conclude that our results aren't all that sensitive to using a uniform prior vs. a normal prior, assuming that the priors are sufficiently wide.

Remember that we can compute F_1 and F_2 are via `ecdf()` on the draws.

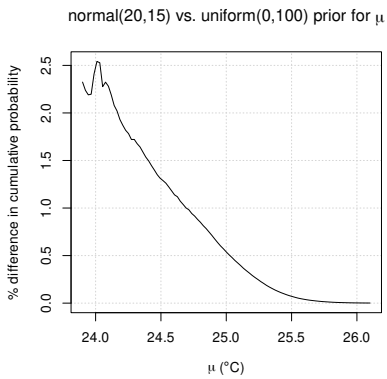


Figure 18.8: The percent difference between the cumulative probability function for μ arising from uniform(0,100) and normal(20,15) priors on μ. To reduce random bumpiness as much as possible, the graph was generated using the combined draws from 10 runs of `stan` with `iter=1e7` and `chains=16`, which requires 16 cpu cores and around 60GB RAM. Using a lower number for `iter` and `chains` will result in a bumpier graph, but will not change the general size of the percent differences too much until you get to the left tail, where probabilities are low (meaning few draws, meaning less reliable numbers).

Choosing A Narrow, Off-Center Normal Prior

What if we artificially choose an overly-narrow density that is pretty far-off from the data? For example, if we compare $\mu \sim \text{uniform}(0, 100)$ and $\mu \sim \text{normal}(10, 5)$, now the means of the posterior densities differ by 0.2% (25.028 vs 24.979). The difference is bigger, but still pretty negligible. However, when computing probabilities, the differences become sizable. Fig. 18.9 shows the %-difference between the cumulative posterior probability for the two priors. We see that the cumulative posterior probabilities differ by very large percentages, approaching 100% difference near

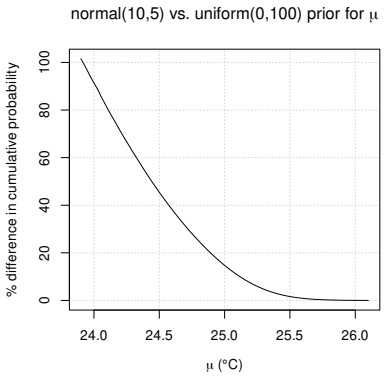


Figure 18.9: The percent difference between the cumulative probability function for μ arising from uniform(0,100) and normal(10,5) priors on μ.

the left-tail. As already stated, if we choose a prior that is overly narrow for no reason, then our results can vary significantly.

Conclusion

In conclusion, trying out different priors in this way suggests that our results are stable across various uninformed or weakly-informed priors, assuming they aren't *too narrow*. We are thus reasonably confident in our analysis this chapter.

Going forward, however, we should be aware that some statistical models can be quite sensitive to the choice of prior, and as a result it's always a good idea to at least do a crude exploration of the model sensitivity of the type considered here.

18.8 Exercises

A student studying the Western Meadowlark (*Sturnella neglecta*) wants to learn more about the mass of these birds. She picks 17 Western Meadowlarks at random and weighs them, obtaining the following data (in grams): 107.3, 106.0, 93.1, 102.2, 117.2, 117.4, 97.3, 104.7, 81.2, 86.5, 87.5, 106.6, 86.1, 101.7, 97.4, 122.4, 86.4.

1. Assume the bird masses are statistically modeled with a normal density of mean μ and standard deviation σ. We wish to find out more about μ and σ. Specify a statistical model appropriate to a Bayesian analysis.
2. Write an R script (and associated Stan file) that computes the following:
   i) Point estimates of μ and σ.
   ii) 50, 80, 95, and 99% ETI and HDI for μ and σ.
3. What is the standard deviation s of the marginal posterior density $\rho(\sigma|D)$? What is the meaning of the standard deviation of a standard deviation here?
4. Create well-labeled plots of the marginal posterior densities for μ and σ.
5. Create well-labeled plots of the cumulative posterior function for both μ and σ. Explain how one can use this graph to estimate credible intervals.
6. Conduct a rudimentary sensitivity analysis by choosing two additional priors for μ and making plots of the type shown in Figs. 18.8 and 18.9.
7. In the text we considered only the sensitivity of the results to the choice of prior for μ. How sensitive are the results to the choice of prior for σ?

18.9 Appendix - Computational Details

This appendix is optional reading on how Bayes' theorem works with continuous data from a normal density. One issue is that μ and σ are not confined to be between 0 and 1. In fact, generally μ can vary between $-\infty$ and $+\infty$, and σ can be between 0 and $+\infty$. This creates two issues: i) we can't specify a uniform prior over the whole possible range of either parameter, since a uniform density over an infinitely large interval is just zero everywhere, and ii) when formulating hypothesis H_i as in Ch. 15, no matter how large we make them, there are still an infinite number of them. And we can't have a computer do an infinite number of things.

The solution to the first problem is to either not use uniform priors, or to use physically realistic cut-offs on the parameters. If we choose the latter, we also solve the second problem of infinite hypotheses.

Following the discussion in Sec. 18.2, it is safe to assume that $\mu < 0°\text{C}$ and $\sigma < 100°\text{C}$. Of course we could actually specify a much smaller range of possible values for the parameters, but this is good enough for our purposes. With the restrictions on μ and σ in place, we can now fully specify our question, possible answers (i.e. hypotheses), data, and statistical models:

Question:	What are μ and σ?
Possible Answers:	$\underbrace{\mu \in (0, 0.1) \text{ and } \sigma \in (0, 0.1)}_{H_{1,1}}$
	$\underbrace{\mu \in (0.1, 0.2) \text{ and } \sigma \in (0, 0.1)}_{H_{2,1}}$
	$\vdots$
$\delta = 0.1$	$\underbrace{\mu \in \Big(\delta \cdot (i-1), \delta \cdot i\Big) \text{ and } \sigma \in \Big(\delta \cdot (j-1), \delta \cdot j\Big)}_{H_{i,j}}$
	$\vdots$
	$\underbrace{\mu \in (99.9, 100) \text{ and } \sigma \in (99.8, 99.9)}_{H_{1000,999}}$
	$\underbrace{\mu \in (99.9, 100) \text{ and } \sigma \in (99.9, 100)}_{H_{1000,1000}}$
Data:	$D = T_k \quad , \quad k = 1, 2, \ldots, N$
Statistical Models:	$T \sim \text{normal}(\mu, \sigma) \quad \leftarrow \text{likelihood} = \mathbb{P}(D\|H_{i,j})$
	$\mu \sim \text{uniform}(0, 100) \quad \leftarrow \text{priors} = \mathbb{P}(H_{i,j})$
	$\sigma \sim \text{uniform}(0, 100)$

To see how Bayes' theorem works, let's for the moment focus on a specific hypothesis, say $H_{204,437}$, which is the hypothesis that $\mu \in (20.3, 20.4)$ and $\sigma \in (43.6, 43.7)$. Define $\delta = 0.1$ as our interval widths for the hypotheses. Let us adopt the notation that $\mu_{204} = 20.3$ and that $\mu_i = (i-1) \cdot \delta$. Similarly, let $\sigma_{437} = 43.6$ and

$\sigma_j = (j-1)\cdot\delta$. Bayes' Theorem says that

$$\mathbb{P}\left(H_{204,437}|D\right) = \frac{\mathbb{P}\left(D|H_{204,437}\right)\cdot\mathbb{P}\left(H_{204,437}\right)}{\mathbb{P}(D)}. \tag{18.4}$$

As before, we can reformulate the prior and posterior distributions as densities to obtain

$$\rho\left(\mu_{204},\sigma_{437}|D\right) = \frac{\mathbb{P}\left(D|\mu_{204},\sigma_{437}\right)\cdot\rho\left(\mu_{204},\sigma_{437}\right)}{\mathbb{P}(D)}. \tag{18.5}$$

We have to be careful about interpreting the likelihoods. We are now dealing with continuous data, and the probability of getting any specific value T from a normal density is zero. The same is true for any sequence of specific values T_k. Thus, it would appear that the likelihood $\mathbb{P}(D|H)$ is always zero, which makes the whole calculation fall apart. The solution to this problem is simple: let's pick some small number ϵ, and interpret a likelihood $\mathbb{P}(T|H)$ as the probability of finding a temperature in the range $(T, T+\epsilon)$. In other words, when calculating the likelihood for a single datum T_k we're really calculating

$$\mathbb{P}\Big(T_k \in (T, T+\epsilon)\Big|\mu,\sigma\Big) = \rho\Big(T\Big|\mu,\sigma\Big)\cdot\epsilon$$

where now we are dealing with a probability density. Note that the same factor of ϵ will also show up in each term of the denominator $\mathbb{P}(D)$. These factors of ϵ will cancel, and we are finally left with

$$\rho\left(\mu_i,\sigma_j|D\right) = \frac{\rho\left(D|\mu_i,\sigma_j\right)\cdot\rho\left(\mu_i,\sigma_j\right)}{\rho(D)}. \tag{18.6}$$

so that now all the pieces are density functions.

Note that we're using the same symbol ρ for all the densities, just as we have used the same symbol $\mathbb{P}$ for all probabilities. We are NOT implying everything symbolized by ρ is the same mathematical function; rather, we're just using the symbol as a general reminder that we're dealing with densities. It's mathematically sloppy notation, but using different names for all the different functions makes for extremely cumbersome notation.

To gain better insight into how specific values of the posterior density are computed, let's return to $H_{204,437}$, corresponding to $\mu_{204} = 20.3$ and $\sigma_{437} = 43.6$. We have

$$\rho\big(\mu_{204},\sigma_{437}\big|D\big) = \frac{\rho\big(D\big|\mu_{204},\sigma_{437}\big)\cdot\rho\big(\mu_{204},\sigma_{437}\big)}{\rho(D)}. \tag{18.7}$$

The use of uniform priors, each ranging from 0 to 100, means $\rho(\mu_i,\sigma_j) = 1/(100^2) = 10^{-4}$. Remembering that the data D is a list of N temperatures T_k, the product rule tells us that the likelihood in the numerator is

$$\prod_{k=1}^{N} \text{normal}\left(T_k,\mu_{204},\sigma_{437}\right). \tag{18.8}$$

where $\prod$ indicates multiplication (in a manner analogous to how Σ indicates summation). Finally, the denominator $\rho(D)$ is

$$\sum_{i=1}^{1000}\sum_{j=1}^{1000}\left[\left(\prod_{k=1}^{N}\text{normal}\left(T_k,\mu_i,\sigma_j\right)\right)\cdot\rho(\mu_i,\sigma_j)\right]. \tag{18.9}$$

The denominator is handled easily by software such as Stan, even for choices of δ smaller than our choice of 0.1. We can also write these formulas in terms of arbitrary values of μ and σ, i.e. in terms of the overall posterior density, as

$$\rho\big(\mu,\sigma\big|D\big) = \frac{\rho\big(D\big|\mu,\sigma\big)\cdot\rho\big(\mu,\sigma\big)}{\rho(D)}. \tag{18.10}$$

While we could compute $\rho(\mu,\sigma|D)$ using the `prod()`, `sum()`, and `dnorm()` functions in R, for more complicated types of analysis it is usually *much* easier to simply specify a statistical model and let a program like Stan do the work.

CHAPTER 19

Comparing More Than Two Groups

When comparing more than two groups, the underlying Bayesian methods are similar to that of Ch. 17. We don't introduce any new statistical ideas in this chapter. Instead, we will introduce three new features of R / Stan that will expedite comparisons among a larger number of groups. The new ideas are *matrices*, *for loops*, and the R function `lapply()`. We will apply these three features in comparing three different groups of data that are modeled as draws from normal densities.

19.1 Data & Models

Corn was planted in 3 different fields using 3 different fertilizers. Aside from fertilizer type, the fields are considered to be roughly equivalent: the corn seed, soil, sunlight, irrigation, etc. is supposedly very similar. Ninety days after planting the corn a farmer measures the height of 20 randomly selected plants in each field. These measurements are shown in Tab. 19.1. So which fertilizer is the most effective?

h_A	h_B	h_C
89.6	153.4	140.3
97.4	115.4	91.2
102.6	150.4	107.2
107.7	101.9	77.9
146.4	106.9	126.1
85.0	109.6	63.3
143.1	156.2	111.9
99.3	111.6	132.0
111.6	119.7	99.8
129.7	103.2	109.0
113.6	148.3	144.5
133.3	133.5	106.8
144.5	136.3	91.2
89.7	126.2	142.0
96.9	145.5	125.1
154.3	150.1	114.4
124.6	141.7	94.0
132.6	75.3	116.0
155.1	101.5	141.5
102.6	103.2	140.4

Table 19.1: Corn Height In Inches

When considering potentially larger numbers of groups, it is more convenient to subscript data with numbers. Hence, instead of h_A we will write h_1, etc. We will assume that the h_1 data is described by a normal density, and likewise for h_2 and h_3. In other words, our likelihoods are

$$\begin{aligned} h_1 &\sim \text{normal}\,(\mu_1, \sigma_1) \\ h_2 &\sim \text{normal}\,(\mu_2, \sigma_2) \\ h_3 &\sim \text{normal}\,(\mu_3, \sigma_3) \end{aligned}$$

or just $h_i \sim \text{normal}\,(\mu_i, \sigma_i)$. Note that the subscript i refers here to the field, not to the individual datum values. While we won't do so here, if you wanted to label the individual datum values in field i, you could use a 2nd subscript j (e.g. something like $h_{i,j}$ or $h_{j,i}$) to indicate the j^{th} datum value in field i.

We also need to choose priors for each of the six parameters in the likelihoods. As usual, we express our ignorance via uniform densities. What are reasonable bounds on the uniform priors? Well, clearly we need $\mu > 0$ for all three fields. To establish an upper bound on the μ, we interview a number of farmers and learn that they have never seen a corn plant higher than around 20 ft (240 inches). Thus, we'll pick the priors

$$\begin{aligned} \mu_1 &\sim \text{uniform}\,(0, 240) \\ \mu_2 &\sim \text{uniform}\,(0, 240) \\ \mu_3 &\sim \text{uniform}\,(0, 240) \end{aligned}$$

As per the discussion of Sec. 18.7, if we choose too narrow of a prior, the final results could be substantially affected. It would be worthwhile to conduct a sensitivity analysis, by changing the priors (perhaps to something wider, like 300 or 400 inches) and seeing how much the results change. As it turns out, the results don't change (to within 3 significant figures), and 240 is plenty wide.

For bounds on the σ, by definition the standard deviation is positive, meaning $\sigma > 0$. For an upper bound, we again talk with the farmers, and learn that it is typical for all the corn plants to be within 2 or 3 feet of each other. To be extra safe, we'll go with an upper bound of 10 ft (120 in). Thus we'll pick the priors

$$\begin{aligned}\sigma_1 &\sim \text{uniform}\,(0, 120)\\ \sigma_2 &\sim \text{uniform}\,(0, 120)\\ \sigma_3 &\sim \text{uniform}\,(0, 120)\end{aligned}$$

With our models in place, we can now use R and Stan to conduct the Bayesian analysis.

19.2 `rstan` Implementation

Suppose that the data has been made available in a comma-separated-value (.csv) file named `corn.csv`. The first few lines of the file would look like:

```
"hA","hB","hC"
89.6,153.4,140.3
97.4,115.4,91.2
102.6,150.4,107.2
...
```
(R.19.1)

We can import the data via

```
1  h.df <- read.csv("corn.csv")
```
(R.19.2)

This creates a data frame `h.df` with columns for h_A, h_B, and h_C. We cannot pass a data frame to Stan, however. Instead, we must pass the data either as individual vectors (one for each field), or as a `matrix`, which is essentially just a data frame that can only contain numbers (no qualitative data). We convert from a data frame to a matrix via the `data.matrix()` function. We will also count the number of rows and columns, in order to pass this info to Stan:

Note that a variable name like `N.plants` is okay in R, but not in Stan, since periods aren't allowed. However, underscore symbols are allowed in both R and Stan, which is why we chose the name `N_plants`, etc.

```
2  h <- data.matrix(h.df)
3  N_plants <- nrow(h)
4  N_fields <- ncol(h)
```
(R.19.3)

We will now write a Stan file that is going to receive all of the height information `h` as a matrix. We first specify our data section as

```
data {
   int N_plants;
   int N_fields;
   real h[N_plants,N_fields];
}
```
(R.19.4)

where we have specified that Stan will be getting a matrix `h` that has `N_plants` rows and `N_fields` columns. We now specify the parameters. Instead of having 3 separate specifications for μ and 3 for σ, we instead collect them into vectors of length 3:

(R.19.5)
```
parameters {
   real<lower=0,upper=240> mu[N_fields];
   real<lower=0,upper=120> sigma[N_fields];
}
```

Our model code section will be

(R.19.6)
```
model {
   for (i in 1:N_fields) {
      h[,i] ~ normal(mu[i],sigma[i]);   //likelihood
      mu[i] ~ uniform(0,240);           //prior
      sigma[i] ~ uniform(0,120);
   }
}
```

where `h[,i]` follows normal R notation: it says give me all the rows of the i^{th} column of the matrix. The `for (i in 1:N_fields)` instruction says "do the lines in the braces `{ }` for $i = 1$, then $i = 2$, then $i = 3$". Note that the above Stan file will work just as well whether we send 3 groups or 100 groups!

Coming back to the R script, we can now call `stan()` and extract the posterior draws:

(R.19.7)
```
5  library(rstan)
6  corn.stan <- stan(file="corn.stan",
7                    data=list(N_plants=N_plants,
8                              N_fields=N_fields,h=h),
9                    iter=2e4,
10                   chains=4, cores=4)
11 mu <- extract(corn.stan)$mu
12 sigma <- extract(corn.stan)$sigma
```

Because there are three μ parameters (one for each field), `mu` is actually a matrix, where each column gives the vector of draws for the corresponding μ. Thus `mu[,1]` gives the draws for μ from the first field, etc. And likewise for `sigma`. We could keep the draws in a matrix, but instead we will convert this matrix back to a data frame using the `as.data.frame()` function. We'll also give the data frame some column names.

(R.19.8)
```
13 mu <- as.data.frame(mu)
14 colnames(mu) <- c("mu1","mu2","mu3")
15 sigma <- as.data.frame(sigma)
16 colnames(sigma) <- c("sigma1","sigma2","sigma3")
```

Line 14 in (R.19.8), which mirrors what we did back in (R.6.15), appears to violate our convention that only variable names ever show up on the left of an assignment operator. Here we are seemingly storing something to the output of a function. Well, `colnames()` is something known as a *replacement function* in R lingo. Luckily for us, the complicated details are hidden from view, and we won't need to worry about it. Incidentally, the square brackets in a statement like `x[2] <- 4` are also replacement functions.

Before we proceed with the analysis, we will pause to talk about the very powerful `lapply()` function in R.

19.3 Lists and `lapply()`

19.3.1 What Is A List?

A `list` in R is like a vector, except the elements don't have to be the same type of thing. We see this in action in line 7 of (R.19.8) above: the first two elements

of the list are single integers, while the third element is a matrix.

In fact, *data frames are actually lists*, where each element of the list is a vector of data representing a column. The list is a very common type of object in R. As such, becoming more comfortable with list operations opens up a lot of power and convenience in using R.

19.3.2 Accessing Elements Of A List

When accessing elements of a vector, one uses a pair of single square brackets. For example, if `x` is a vector, then `x[3]` accesses the 3rd element of `x`. This notation only works for vectors, and not for lists.

In R, there are multiple ways to access the element of a list. For example, to get the first column in the data frame `mu`, we could use the element name with dollar-sign notation, `mu$mu1`. If the list is specifically a data frame we could also use bracket notation as `mu[,1]`.

However, there is a more general way to access elements of any type of list in R: use double square brackets. Hence, `mu[[1]]` also accesses the first element (i.e. column) of our list (i.e. data frame). We will use the double square bracket notation shortly.

19.3.3 `lapply()`

One of the most useful all-purpose functions in R is `lapply()`. The basic idea of the function is simple: apply the same operation or function to each element in a list. The syntax is

$$\texttt{lapply}\left(\boxed{\begin{array}{c}\text{name of}\\ \text{list}\end{array}}\ ,\ \boxed{\begin{array}{c}\texttt{FUN =}\\ \text{function to apply to list elements}\end{array}}\right) \tag{19.1}$$

The function you specify via the `FUN` parameter can either be a built-in function, or a function you specify. If it is a simple built-in function with only one argument, like `mean()`, you just set `FUN=mean`. If it is function that requires multiple arguments, or is not a built-in function, you must use `FUN=function(x)` followed by whatever your function does, and where `x` can be any variable of your choosing. For example, say we want to compute the 0.10 and 0.90 quantiles for the posterior draws for μ. We can use `lapply()` as

```
mu.quants <- lapply(mu,
                    FUN=function(x) quantile(x, c(0.1,0.9) )
                    )
```
(R.19.9)

In fact, anything that you can accomplish with `lapply()` can be done with a `for` loop, though there may be differences in execution time.

The way this works is that `lapply()` essentially loops through the list. It starts by picking out the first element of the list, i.e. the vector of posterior draws for μ_1. This is the first value of `x` evaluated by the function. After finding the quantiles for the first element of the list, `lapply()` moves on to the next element of the list, etc. The result `mu.quants` is itself a list. Hence, typing `mu.quants[[2]]` will return the quantiles for the 2nd element of the list (the draws for μ_2), etc.

We will further illustrate the use of `lapply()` multiple times in the remainder of our analysis this chapter.

19.4 Point Estimates

To get the mean point estimate for each posterior, we could take the `mean()` of each column. However, we can also use `lapply()` on the data frames of the draws. The code

(R.19.10)
```
mu.means <- lapply(mu,FUN=mean)
sigma.means <- lapply(sigma,FUN=mean)
```

returns lists containing the means for the three μ posteriors and the three σ posteriors. The resulting point estimates are:

$$\begin{array}{ll} \mu_1 = 118.0 & \sigma_1 = 24.6 \\ \mu_2 = 124.5 & \sigma_2 = 24.4 \\ \mu_3 = 113.7 & \sigma_3 = 24.7 \end{array}$$

Comment 19.1 – MAP Estimates (Optional Reading)

If you need MAP estimates for the parameters, i.e. the estimates at the peak of the joint posteriors, you could issue the code

```
m <- stan_model(file="corn.stan")
corn.MAP <- optimizing(m,data=list(N_plants=N_plants,
                                   N_fields=N_fields,h=h))
```

to find that

$$\begin{array}{ll} \mu_1 = 118.0 & \sigma_1 = 22.4 \\ \mu_2 = 124.5 & \sigma_2 = 22.2 \\ \mu_3 = 113.7 & \sigma_3 = 22.5 \end{array}$$

We see that the MAP estimates for the μ_i agree with the mean estimates (to within 4 significant figures), but the MAP estimates for the σ_i are about 10% smaller than the mean estimates. Again, this is due to the larger asymmetry of the posteriors with regards to σ_i.

19.5 Violin Plots

As mentioned in Sec. 17.2, the violin plot nicely combines the posterior density and the box plot into a single figure. We could follow the same R code from Sec. 17.2, but instead we'll make the code more efficient for larger numbers of groups.

First, it will be nice to have a vector of labels of the form μ_i for the x-axis. In order to render greek letters and subscripts, we again use the `expression()` function:

(R.19.11)
```
posterior.labels <- c(expression(mu[1]),expression(mu[2]),
                      expression(mu[3]))
```

Comment 19.2 – Autolabeling (Optional Reading)

If there are a large number of different groups, it would be nice to have a more efficient method of creating these labels. We will need to explain a few new ideas in the code:

```
posterior.labels <- as.expression(
   lapply( 1:N_fields, FUN=function(i) bquote(mu[.(i)]) )
)
```

The list we're sending to `lapply()` is `1:N_fields`, which is just the numbers 1,2,3. Our function uses `bquote()`, which treats everything inside as a string of text, *except* that anything prefaced by a period and enclosed by parenthesis will be replaced by the actual value of the variable. Hence, inside the `bquote()` function, the symbols `.(i)` will be replaced by the current value of `i` inside the loop of the `lapply()` function. If we don't do it this way, we'll just end up with a literal letter "`i`" for all the subscripts. Finally, everything is enclosed inside an `as.expression()` function, in order to typeset the greek letter and the subscript properly. We must use `as.expression()` rather than `expression()`, because if we use the latter R will assume we literally want to print the letters "`lapply(...`" instead of executing the commands.

Now let's move on to laying out the plot. In (R.17.5) we decided to center the first violin plot at $x = 0.5$, and then place the next plot one unit to the right at (approximately) $x = 1.5$. If there were another plot, we could put it at $x = 2.5$, etc. The last plot would be at $x = N_{fields} - 0.5$.

```
21  x.locations <- seq(from=0.5, to=N_fields-0.5, by=1)
```
(R.19.12)

Next we create an empty plot window and place a grid and customized x-axis:

```
22  library(vioplot)
23  plot(x=NA,y=NA,
24       main=expression("Violin Plots of "~mu~"Posteriors"),
25       xlim=c(0,N_fields),
26       ylim=range(mu),
27       xlab="",
28       ylab="Height (inches)",
29       xaxt="n")
30  grid(nx=NA,ny=NULL)
31  axis(side=1,at=x.locations,labels=posterior.labels)
```
(R.19.13)

Finally we add a violin plots for each field using a `for` loop (see Comment 7.1). We will automatically generate a vector of different colors using `rainbow()`:

```
32  col.vec <- rainbow(N_fields)
33  for (i in 1:N_fields) {
34     vioplot(mu[,i], at=x.locations[i], add=TRUE, col=col.vec)
35  }
```
(R.19.14)

The resulting plot is shown in Fig.19.1. Note that this plot code would work without modification for larger numbers of groups.

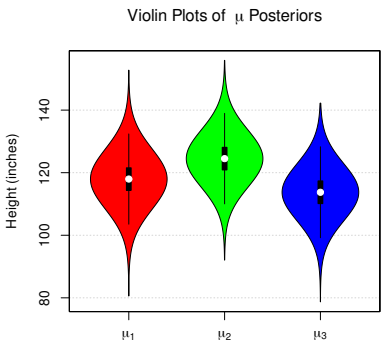


Figure 19.1: Violin plots of posterior draws for μ_1, μ_2, and μ_3. Note that if you would rather have the plots be horizontal, set the `horizontal` parameter to `TRUE` when calling `vioplot()`, and adjust the axes etc. as appropriate.

It appears as though field (fertilizer) #2 has the most growth, as we can see from the boxplots that its 50% credible interval is almost completely above that of the other fields. As usual, to be more quantitative, we'll compute some credible intervals and probabilities.

19.6 Probability and Interval Calculations

There are many different approaches one could take to facilitate comparisons. Here we are going to follow the analysis of Sec. 17.4 and consider draws of the *difference* between the μ_i. Let's consider a data frame consisting of of the difference between μ_2 and μ_i draws, since fertilizer #2 seems to be in the lead:

(R.19.15)
```
dmu <- lapply(mu,FUN=function(x) mu[,2]-x)
```

were this time when using `lapply()` we defined our own function. The first column of `dmu` will contain the draws for $d\mu_{2,1} \equiv \mu_2 - \mu_1$, the second column will contain the draws for $d\mu_{2,2} \equiv \mu_2 - \mu_2$ and will hence be all zeros, and the third column will contain the draws for $\mu_2 - \mu_3$.

We can make quick work of computing, say, 95% HDIs for the $d\mu$ using `lapply()` on the data frame of draws `dmu`:

(R.19.16)
```
library(HDInterval)
dmu.HDI <- lapply(dmu,FUN=function(x) hdi(x,0.95))
```

to obtain

$$\begin{aligned} d\mu_{2,1} \equiv \mu_2 - \mu_1 \quad &: \quad HDI_{0.95} = (-9.0, 22.0) \\ d\mu_{3,1} \equiv \mu_2 - \mu_3 \quad &: \quad HDI_{0.95} = (-4.8, 26.3) \end{aligned}$$

Remember the meaning here: there is a 95% probability that $\mu_2 - \mu_1$ is between -9.0 inches and 22.0 inches. In other words, with 95% probability, the growth of corn with fertilizer #2 is as much as 22 inches higher than that of fertilizer #1, but also possibly 9 inches less, etc. The bulk of the credible intervals are positive, again suggesting $\mu_2 > \mu_1$ and $\mu_2 > \mu_3$.

Following Ex. 17.2, we can compute $\mathbb{P}(\mu_2 > \mu_1)$ and $\mathbb{P}(\mu_2 > \mu_3)$, though here we will use `lapply()`. First we compute the cumulative probability functions for the posteriors in `dmu`:

(R.19.17)
```
dmu.ecdf <- lapply(dmu,FUN=ecdf)
```

Next we will use `lapply()` in a different way, by feeding it a list of *indices* 1,2,3:

Note that in (R.19.14), instead of a `for` loop we could have used this trick of using `lapply()` with a list of indices.

Again, the theme of the chapter is writing efficient R / Stan code if there are a large number of groups. For only 2 or 3 groups, code like (R.19.18) is a bit overkill.

(R.19.18)
```
probs <- lapply(1:N_fields,
                FUN = function(i) 1-dmu.ecdf[[i]](0)
                )
```

which returns

$$\mathbb{P}(\mu_2 > \mu_1) = 80\% \tag{19.2}$$
$$\mathbb{P}(\mu_2 > \mu_3) = 92\% \tag{19.3}$$

Of course, we also want to know *by how much* is fertilizer #2 better than the others. One nice way to summarize this information is a plot like Fig. 17.5. As an exercise, see if you reproduce the plot in Fig. 19.2 below. A similar figure can also be made for $d\mu_{2,3}$, of course.

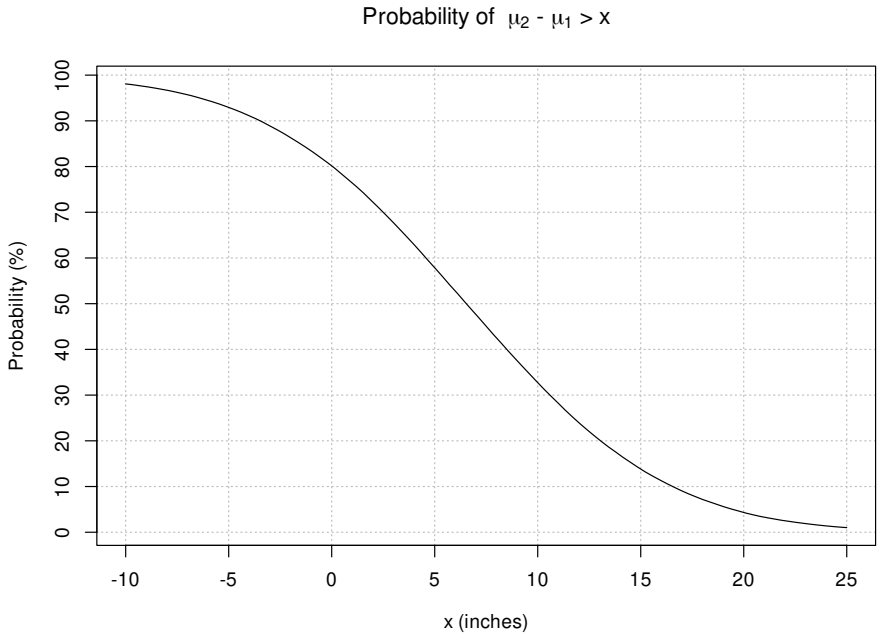


Figure 19.2: The probability that $d\mu_{2,1} \equiv \mu_2 - \mu_1 > x$.

Fig. 19.2 is far more informative than Eq. (19.2). We can immediately confirm from the graph that $\mathbb{P}(\mu_2 > \mu_1 + 0) = \mathbb{P}(d\mu_{2,1} > 0) = 80\%$. But we also see that $\mathbb{P}(\mu_2 > \mu_1 + 10) \approx 33\%$. There is thus only a 33% probability that the extra height due to fertilizer #2 (compared to fertilizer #1) is more than 10 inches.

19.7 So Which Fertilizer Is Most Effective?

Our original question was

> **"Based on the data, which fertilizer is most effective?"**

At first glance this seems like a multiple-choice question, i.e. apparently the answer is either fertilizer #1, #2, or #3. This is not the case for scientific questions. In actuality we are dealing with noisy data and other types of uncertainty. Thus, a valid answer *MUST* involve some type of probability statement that quantifies our uncertainty. Here are some valid answers to our original question:

> "There is an 80% probability that fertilizer #2 is more effective than #1, and a 92% probability it is more effective than #3."

or

> "The probability that fertilizer #2 produces x more corn height than #1 is shown in Fig. 19.2, and the probabilities are even higher for comparison with #3."

or

> "The 95% highest density credible interval for $\mu_2 - \mu_1$ is $(-9, 22)$ inches and for $\mu_2 - \mu_3$ is $(-5, 26)$ inches."

And these types of answers are the best we can do. If we want even more certainty that fertilizer #2 is the best, the only option is to collect more data.

19.8 So Which Fertilizer Should We Use?

The question of which fertilizer to use is a decision question, not a scientific question. This *IS* a multiple choice question.

If everything else about the fertilizers is equal (price, ease of application, environmental impact, etc.) then clearly #2 is our best choice. However, those other factors are seldom equal. For example, what if fertilizer #2 costs 25% more than the other two. Is the extra growth worth it? After all, as shown in Fig. 19.2, there is only about a 33% probability that the difference in height from fertilizer #1 is more than 10 inches. How much extra yield in actual corn results? For that matter, are *any* of the fertilizers worth it? After all, our data doesn't involve a control group, so we don't know the corn height without fertilizers.

What if fertilizer #2 only costs 2% more money than the others, but it has a much more negative environmental impact? Even if using this fertilizer resulted in a net gain in profits, is it worth it? How much value is placed in environmental impact?

These kinds of questions clearly go beyond our Bayesian analysis. Other types of costs must be evaluated, and human judgments must be used. The math we're doing in this chapter can't make the decision for us, though it can certainly *inform* our decision.

19.9 Exercises

a	b	c	d
235	181	254	166
233	173	254	152
209	206	295	137
250	156	199	183
238	207	274	141
204	140	211	128
259	111	242	141
209	197	236	148
237	168	208	144
318	151	252	145

Table 19.2: Lead concentration in ppm in soil from four different areas.

A local government agency has hired you to advise them on a pollution clean-up project. Local technicians have collected 10 soil samples from each of four areas, labeled a, b, c, and d, that they feel may be at risk for unsafe lead pollution. The samples have been analyzed, and the concentration of lead (Pb) in the samples, in ppm, is given in Tab. 19.2. The government has indicated that it would only have clean-up funds for one of the areas.

You are to analyze the data and offer your analysis and recommendations in a report. Your report should look professional (neat, organized, spell-checked, clearly labeled diagrams, etc.), be submitted as a single pdf file, and should contain the following:

1. point estimates and 50, 95, and 99% credible intervals for the μ_i
2. side-by-side violin plots of the posterior draws of each μ_i
3. all pairwise probabilities of the form $\mathbb{P}(\mu_a > \mu_b)$
4. a summary of the EPA guidelines around acceptable Pb levels in soil, as given in Title IV of the Toxic Substances Control Act (TSCA) of 1976, and your assessment of whether each of the four areas a, b, c, d meets these safety guidelines.
5. a concluding section giving your opinion about whether the government should spend its funds on lead cleanup in one of the areas (indicate which one), or rather should save these funds to invest in other environmental initiatives (such as solar-panel installations or river clean-up).
6. an appendix giving your commented R and STAN code

NOTE: While readers of your report may have a science/math background, they may be rusty on their statistics. Thus, you should include brief explanations of the various numbers/plots you give. In other words, briefly explain how to interpret a posterior density plot, a credible interval, etc.

Part IV

Frequentist Methods

CHAPTER 20

Distribution of Sample Means

In Part III we used Bayesian methods to infer the values of parameters (such as μ) for various probability densities. Frequentist methods involve a rather different approach to statistical inference. Many of these methods rely upon the concept of the *sample mean distribution*, which we discuss in this chapter. Before diving in, we pause to briefly discuss the basic difference between the Bayesian and the Frequentist approaches.

20.1 What Does "Frequentist" Mean?

From a Bayesian perspective probability represents a subjective uncertainty about something. For example: a researcher doesn't know the exact value of the mean height μ of some bird species, and she could express this uncertainty by assigning a prior probability. After looking at data, she could then compute a posterior probability density for μ, from which she could compute credible intervals, etc. Put another way, the Bayesian viewpoint uses known data D to assign probabilities to various hypotheses H regarding the unknown value of the parameter μ. You would not ascribe any sort of probability to your data D, because you are looking right at it, and it is *certain*.

From the frequentist perspective, probability is defined as the long-term frequency of something happening. For example, if you imagine flipping a coin a huge number of times, and 30% of the time it comes up heads, then you say there is a 30% probability of getting the outcome "heads". Note that this kind of thought process doesn't work when thinking about a parameter like μ, because μ is never the "outcome" of some event. The outcomes are the *data* that result from some type of experiment or measurement.

Hence, in frequentist terms it makes no sense to ascribe a probability to μ. Instead, frequentist methods assign probabilities to the data D. Every time a sample is drawn this data will be different, but in the long run samples will have certain properties that we can leverage to infer something about the values of parameters like μ.

	Bayesian	**Frequentist**
probability defined as	measure of uncertainty	long-term frequency of an outcome
probability assigned to	hypotheses H	data D

20.2 Samples and Populations

In Part. III we discussed data in terms of draws from a distribution or density. Thus, if our data are 43 heights of a species of plant, we would conceptualize these 43 numbers as being a group of draws from, say, a normal density. We think of the draws as being `samples` from a `population`.

In statistics, the words *sample* and *population* have a precise meaning that is somewhat different from their typical scientific meaning:

- When we say *sample*, we are **not** referring to 43 actual physical plants. Instead, we **are** referring to just the 43 numbers, i.e. heights.
- When we say *population*, we are **not** referring to the total *biological* population of the physical plants actually existing in nature right now. Also, we are **not** referring to the total set of all bird heights existing in nature right now. Instead, we **are** referring to a *statistical* population, meaning all the *possible* plant heights (with some heights more prevalent than others).

To sharpen the the meaning of *statistical population*, consider an endangered species, such as the Sumatran rhino. As of 2021, there were approximately 80 sumatran rhinos known to exist in nature. Hence, the *biological population* is 80. Imagine we measured the weight of all 80 sumatran rhinos, i.e. we measured the weight of every rhino in the biological population. If the weight of the rhinos is described by normal density with mean μ, we still don't know μ with total certainty, even though we measured the weight of every rhino in the entire biological population! The point is that we can't measure every *possibly existing* rhino, and it is this infinite collection of *possible weights*, some more prevalent than others, that comprises the *statistical population*.

20.3 Sample Mean Distribution

Consider the following situation: you are asked to measure the heights of 30 randomly-selected people. You do so, and compute the mean height for your sample data. Your friend is also asked to randomly select 30 people and measure their heights. They obtain their own sample mean. Imagine that this continues, i.e. that a *huge* number of people are each given the task of randomly selecting 30 people, measuring their heights, and computing the sample mean.

What we end up with is a *huge* number of different sample means. This collection of sample means can itself be considered a statistical population. Equivalently, the sample means themselves can be considered as draws from some probability density. In other words, if you go out and find the mean height of 30 randomly selected people, the result is itself random. The mean height you calculate will depend upon the people you happened to select.

Actually, we're being imprecise here. Following the discussion of the previous section, the actual population would be the collection of all *theoretically possible* sample means, not just the sample means one could obtain with currently existing samples. Our *huge number* of different sample means is only an approximation to the true statistical population.

When we talk of the `sample mean distribution` or *density*, or SMD, we refer to the fact that for a given sample size n, we'll end up with a different sample mean depending upon whatever ends up comprising our sample. *We are not here talking about the distribution of values within a given sample.* We are talking about the different possible means we could obtain when we draw a sample.

Moving forward, we'll often use the word *distribution* for both true distributions and also densities. This is a bit sloppy, but is pretty commonplace in the statistics world.

20.4 Sample Means for Normal Data

Consider drawing a sample of a variable x, say $x_1, x_2, \ldots, x_n$, from a normally-distributed population with mean μ and standard deviation σ. It turns out that the sample-mean $\bar{x} = \sum x_i/n$ itself is statistically modeled by a normal density, no matter the size of the sample. If our hypothetical *huge* number of people all report their sample-means, so that we essentially have the entire sample-mean population, what are the mean and standard deviation of this population? Let's first consider, in qualitative terms, what happens as the sample size increases.

20.4.1 Impact of Sample Size on the SMD

First imagine your samples contain only one data point x_1, i.e. your sample size is $n = 1$. Clearly the mean $\bar{x}$ of the sample is just the value of x_1. Hence, the distribution of $\bar{x}$ is identical to the distribution of x, which is just the population distribution. Thus, when the sample size is $n = 1$, the SMD has the same mean and standard deviation as the population distribution.

What if your samples had two data points, x_1 and x_2? We can write each data point as $x_i = \mu + r_i$, where r_i is the residual of the data point from the population mean μ. If $r_i > 0$ then x_i is larger than μ, and if $r_i < 0$ then x_i is smaller than μ. The mean of our sample is then

$$\begin{aligned}
\bar{x} &= \frac{x_1 + x_2}{2} \\
&= \frac{(\mu + r_1) + (\mu + r_2)}{2} \\
&= \frac{2\mu + r_1 + r_2}{2} \\
&= \frac{2\mu}{2} + \frac{r_1 + r_2}{2} \\
&= \mu + \bar{r} \qquad (20.1)
\end{aligned}$$

Using the same type of calculation, it is straightforward to show that Eq. (20.1) is true for any sample size n.

i.e. the sample mean $\bar{x}$ is equal to the population mean μ plus the mean of the residuals. However, these residuals tend to cancel each other. Note that for any given data point x_i there is a 50% probability that $r_i > 0$, and 50% that $r_i < 0$. While there is a 1/4 chance both residuals are negative, and a 1/4 chance both are positive, there is still a 50% probability that one is positive while the other is negative. When the residuals have opposite signs, they tend to cancel each other out (i.e. have a mean close to zero), and hence $\bar{x}$ is likely close to $\bar{\mu}$. Of course, the other 50% of the time the two residuals don't cancel. Still, on the whole, for $n = 2$ we are more likely to find $\bar{x}$ closer to μ than we were for $n = 1$, because at least *some of the time* the residuals partially cancel out.

When the sample size gets bigger and bigger, this cancellation is much more likely to happen. Thinking of each data point as a 50/50 coin-flip, where the outcome is either heads ($r_i > 0$) or tails ($r_i < 0$), we know that as the number of flips gets large, by far the most likely outcome is that the number of heads is close to the number of tails, e.g. there is close to zero chance that there would be something like 80 heads and 20 tails. Hence there is a lot of cancellation, and the mean residual $\bar{r}$ in Eq. (20.1) gets closer and closer to zero as n increases.

The punchline is this: when the sample sizes are large, there is more of an opportunity for residuals to cancel each other out, so that our sample mean $\bar{x}$ is more likely to remain closer to the population mean μ. Thus, for $n > 1$, the standard deviation s_x of the SMD is smaller than the population standard deviation σ. As the sample size n increases, s_x decreases.

While we won't prove it here, it turns out that for samples of size n drawn from a normal distribution with mean μ and standard deviation σ, the SMD is itself normal with mean and sd

$$\bar{x} = \text{mean of sample-mean distribution} = \mu \tag{20.2}$$

$$s_x = \text{standard deviation of sample-mean distribution} = \frac{\sigma}{\sqrt{n}} \tag{20.3}$$

Thus, the SMD has the same mean as the original population density from which the samples are drawn. However, the standard deviation of the SMD is narrower than that of the original population density by a factor of $\sqrt{n}$.

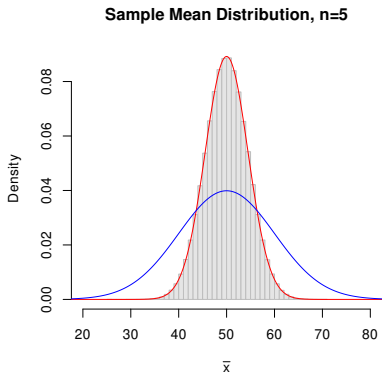


Figure 20.1: Histogram of sample means $\bar{x}$ for $N = 10^6$ samples of size $n = 5$ drawn from a normal density (blue curve) with $\mu = 50, \sigma = 10$. The red curve is a normal density with $\mu = 50$, $\sigma = 10/\sqrt{5}$.

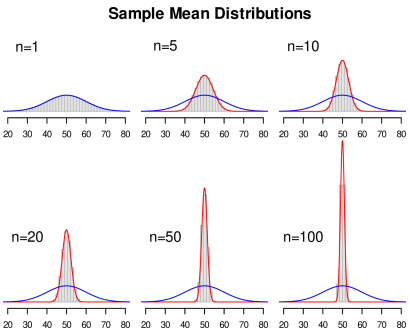


Figure 20.2: Histograms of sample means $\bar{x}$ for $N = 10^7$ samples of size n drawn from a normal density with $\mu = 50, \sigma = 10$, represented by the blue curve. The red curve is a normal density with $\mu = 50$, $\sigma = 10/\sqrt{n}$.

20.4.2 Simulating Sample Means

We can use simulation to verify that s_x narrows as the sample size n increases. Consider size $n = 5$ samples from a normal density with $\mu = 50$ and $\sigma = 10$. We can easily simulate a sample of size n (i.e. n draws) using `rnorm()`, and then compute the `mean()` of this sample:

```
sample.size <- 5
sample <- rnorm(n=sample.size,mean=50,sd=10)
sample.mean <- mean(sample)
```
(R.20.1)

or equivalently

```
sample.size <- 5
sample.mean <- mean(rnorm(n=sample.size,mean=50,sd=10))
```
(R.20.2)

Now let's do the same thing a bunch. In other words, let's draw a bunch of random samples, and compute the mean and sd of each one. We'll store all of the samples means for 1 million samples in a vector called `sample.means`, via the `replicate()` function:

```
number.samples <- 1e6
sample.size <- 5
sample.means <- replicate(n=2number.samples,
                          mean(rnorm(n=sample.size,mean=50,sd=10)))
```
(R.20.3)

We can then plot a histogram of `sample.means`. This histogram is shown in Fig. 20.1, with a normal density ($\mu = 50, \sigma = 10/\sqrt{5}$) overlayed. Note that the sample mean distribution is *narrower* than the original distribution.

As the sample size n gets larger, the width of the sample-mean distribution gets smaller, as shown in Fig. 20.2. In other words, as our sample size n gets larger, the resulting sample mean is increasingly likely to lie very close to the population mean μ. In a hypothetical limit as n approaches infinity, then all sample means $\bar{x}$

will exactly equal the population mean μ, and correspondingly s_x becomes zero, since there is no spread of values for $\bar{x}$.

20.4.3 Standard Normal SMD

It will prove convenient to write Eqs. (20.2) and (20.3) in terms of the standard scores z defined in Eq. (14.3). If $\bar{x}$ is described by a normal density with mean μ and standard deviation $\sigma/\sqrt{n}$, then by definition

$$\frac{\bar{x}-\mu}{\sigma/\sqrt{n}} = z \qquad \text{where} \qquad z \sim \text{normal}(0,1). \tag{20.4}$$

In other words, $(\bar{x}-\mu)/(\sigma/\sqrt{n})$ follows a standard normal or z density.

Example 20.1: Cumulative Sample Mean Probability

This is not the most practical or realistic example, since μ and σ are not typically known ahead of time. The purpose of this example is simply to reinforce the *concept* behind the SMD.

Problem: A normal population has a known mean $\mu = 34$ and standard deviation $\sigma = 7$. If a researcher draws a sample of size $n = 6$ and computes its sample mean $\bar{x}$, what is the probability that $30 \le \bar{x} \le 38$?

Solution: Let $\bar{x}_L = 30$ and $\bar{x}_R = 38$ denote the leftmost and rightmost value of $\bar{x}$ being considered. According to Eq. (20.4), the corresponding values of z are

In this example z is $\pm$ the same number because the values of $\bar{x}_L$ and $\bar{x}_R$ are equal distances to the left and right of μ.

$$\bar{x}_L = 30 \quad \rightarrow \quad z_L = \frac{30-34}{7/\sqrt{6}} \approx -1.40$$

$$\bar{x}_L = 38 \quad \rightarrow \quad z_R = \frac{38-34}{7/\sqrt{6}} \approx +1.40$$

We can then use `pnorm()` to compute the corresponding probability:

$$\begin{aligned}
\mathbb{P}(-1.4 \le z \le 1.4) &= \text{area between } -1.4 \text{ and } 1.4 \\
&= \quad \text{ } \\
&= \Big(\text{area between } -\infty \text{ and } 1.4\Big) \\
&\qquad - \Big(\text{area between } -\infty \text{ and } -1.4\Big) \\
&= \quad \text{ } \quad - \quad \text{ } \\
&= \texttt{pnorm(1.4)} - \texttt{pnorm(-1.4)} \\
&= 0.919 - 0.081 \approx \boxed{0.838}
\end{aligned}$$

This also implies there is $1-0.838 \approx 0.16$ probability that $\bar{x}$ would be outside the specified range.

A common problem in statistics is to make inferences about μ on the basis of the sample mean $\bar{x}$. We'll explore how to do this in the next chapter, but for the time being note that if we don't know μ, then we almost certainly don't know σ. Thus, it will prove very useful to find an equation like Eq. 20.4 that doesn't involve the unknown parameter σ. In order to achieve this, we'll first need to examine the distribution of sample standard deviations.

20.5 The Distribution of Sample Variances

Let's examine the distribution of sample variances s^2 (the square of the sample standard deviation s) computed according to Eq. (3.1):

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 \tag{20.5}$$

$$= \frac{1}{n-1}\sum_{i=1}^{n} r_i^2 \tag{20.6}$$

where $r_i \equiv x_i - \bar{x}$. Now assume that the x_i are drawn from a normal density with standard deviation σ. This means that the residuals r_i also a normal density: it's the same bell-curve as the x_i follow, but merely shifted by $\bar{x}$. Note that this bell curve (normal density) is merely shifted, and not stretched, so that it has the same standard deviation σ. Thus, the *residuals* r_i are themselves normally distributed variables with standard deviation σ.

We will therefore divide both sides of Eq. (20.6) by σ^2 and obtain

$$\frac{s^2}{\sigma^2} = \frac{1}{1-n}\sum_{i=1}^{n} z_i^2 \tag{20.7}$$

where the standard normal variable z_i is defined as

$$z_i \equiv \frac{r_i}{\sigma} = \frac{x_i - \bar{x}}{\sigma}. \tag{20.8}$$

We can multiply Eq. 20.7 by $n-1$ to find that

$$\frac{(n-1)s^2}{\sigma^2} = \sum_{i=1}^{n} z_i^2, \tag{20.9}$$

Even though the summation in Eq. (20.9) extends from 1 to n, only $n-1$ of these z_i are independent, because once we know $n-1$ of the values of z_i, the last value z_n is given via $\sum_{i=1}^{n} z_i = 0$.

where the RHS of this equation is modeled statistically by the χ^2_{n-1} distribution of Sec. 14.5.1, i.e. as the sum of $n-1$ independent standard normal variables squared. If the RHS is modeled by a χ^2_{n-1} density, then so is the LHS. We arrive at the result that

$$\frac{(n-1)s^2}{\sigma^2} = \chi^2_{n-1} \qquad \text{where} \qquad \chi^2_{n-1} \sim \text{chisq}(n-1) \tag{20.10}$$

Thus, $(n-1)s^2/\sigma^2$ is statistically modeled by a chi-squared density with $n-1$ degrees of freedom. We will now combine this result with Eq. (20.4).

20.6 t Distribution of Sample Means

We can solve Eq. (20.10) for σ and substitute this into Eq. (20.4) to obtain (checking this is good algebra practice!)

$$\frac{\bar{x} - \mu}{s/\sqrt{n}} = \frac{z}{\sqrt{\frac{\chi^2_{n-1}}{n-1}}} \tag{20.11}$$

Recalling Eq. (14.7), we recognize the RHS of Eq. (20.11) as a student's t density with $n-1$ degrees of freedom:

$$\frac{\bar{x}-\mu}{s/\sqrt{n}} = t_{n-1} \qquad \text{where} \qquad t_{n-1} \sim \text{student}(n-1) \tag{20.12}$$

Thus, $(\bar{x}-\mu)/(s/\sqrt{n})$ is statistically modeled by a student's t density with $n-1$ degrees of freedom. As we will see later, Eq. (20.12) is an important ingredient in the methods of confidence intervals and null-hypothesis significance testing (NHST).

Comment 20.1 – Don't Solve For μ (Optional Reading)

It is tempting to solve Eq. (20.12) for μ. Doing so results in

$$\mu = \bar{x} - \frac{s}{\sqrt{n}} \cdot t_{n-1} \tag{20.13}$$

which suggests that μ is described by a (shifted and stretched) t-distribution. If this were true, then we could do all kinds of probability calculations for μ, just like we did in our Bayesian analyses.

In a Bayesian analysis it actually *is* meaningful to solve for μ in this way, but we usually don't, simply because most of the time there is no need to worry about the sample mean distribution. We just model the individual data points with a likelihood function, and then use Bayes' theorem to compute a posterior density for μ.

However, in a frequentist analysis it is not meaningful to solve Eq. (20.12) for μ in this way. Remember that in the frequentist viewpoint, we can only assign probabilities to data, not to parameters like μ. Thus, in a frequentist analysis it is meaningless to say that μ is modeled as a t distribution.

This all raises the question: How exactly *do* we use Eq. (20.12) to try and infer something about μ? This is actually a significant problem with the frequentist approach, owing to the fact that probability is assigned to the thing we know for certain (our data) rather than the thing we don't know for certain (the parameter μ). One approach to the frequentist inference of μ is that of the *confidence interval*, which we discuss in the next chapter.

It is important at this point to realize that Eq. (20.12) is only valid for sample data drawn from a normal density. However, we often have no clue about the form of the population density. Sometimes we even have strong evidence that it's definitely *not* normal. Let's examine the extent to which Eq. (20.12) is approximately true in such cases.

Bayesian methods don't evade this issue. While it is true that Bayesian analyses don't usually bother with trying to model the sample mean distribution, one still needs to specify the likelihood $\mathbb{P}(D|H)$, and it is often just a guess that this is a normal distribution.

20.7 SMD For Non-Normal Data

If sample data is drawn from a non-normal density, then Eq. (20.12) isn't true. Happily, there are various theorems (such as the well-known *central limit theorem*) in statistics saying that as n gets bigger and bigger, Eq. (20.12) becomes more and more true for almost *any* density. But how big does n have to get before Eq. (20.12) is "mostly valid"?

A frequently invoked bit of conventional wisdom says $n = 30$ is enough, but this is demonstrably false. For example, Fig. 20.3 shows the results of a simulation in which $N = 5e7$ samples of size $n = 30$ are drawn from a standard lognormal density, and the LHS of Eq. (20.12) is computed for each sample. The histogram

A lognormal density arises when the logarithm of a random variable $\log(x_i)$ follows a normal density.

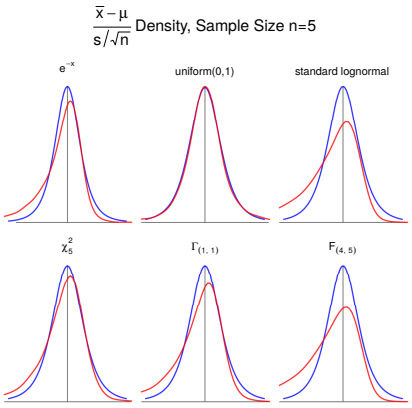


Figure 20.4: Density (red line) of $\frac{\bar{x}-\mu}{s/\sqrt{n}}$ for $N = 1e5$ samples of size $n = 5$ drawn from six different density functions. The blue line is the t_{n-1} density function.

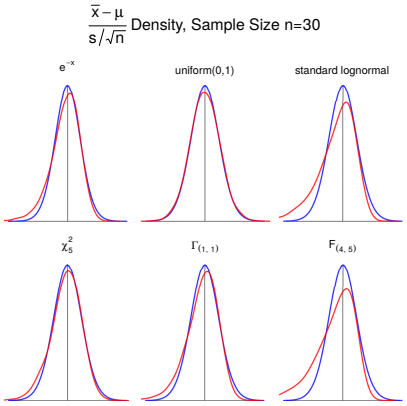


Figure 20.5: Density (red line) of $\frac{\bar{x}-\mu}{s/\sqrt{n}}$ for $N = 1e5$ samples of size $n = 30$

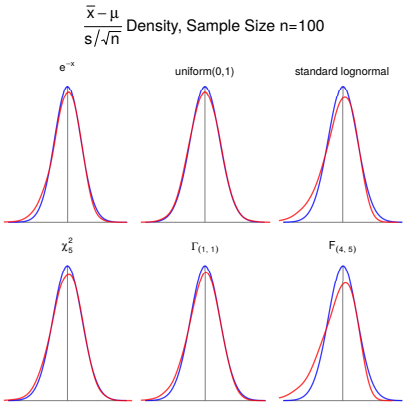


Figure 20.6: Density (red line) of $\frac{\bar{x}-\mu}{s/\sqrt{n}}$ for $N = 1e5$ samples of size $n = 100$

of the results is shown with a t_{29} density overlayed, and we can see that they noticeably disagree. When we discuss NHST in Ch. 22 we'll see that conclusions

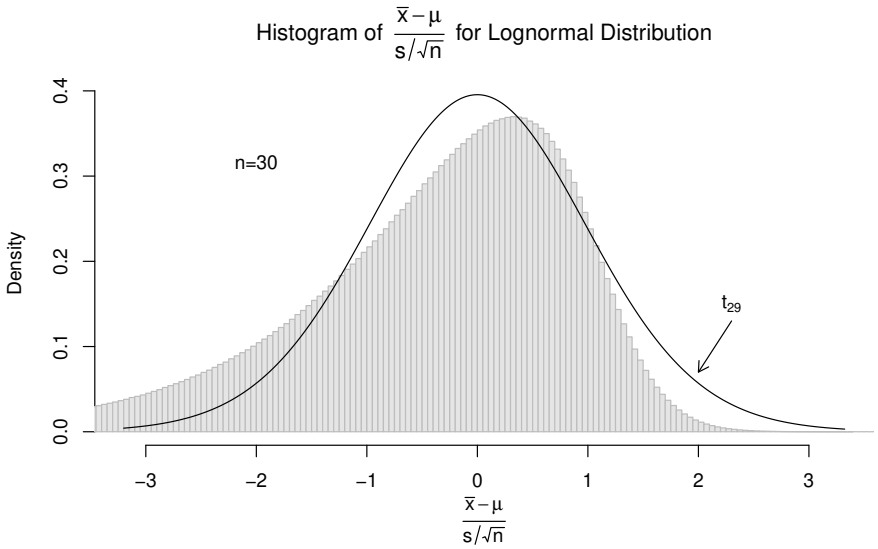


Figure 20.3: Histogram of $\frac{\bar{x}-\mu}{s/\sqrt{n}}$ for $N = 5e7$ samples of size $n = 30$ drawn from a standard lognormal density. The black line is the t_{n-1} density function.

are very sensitive to what's going on in the tails, and Fig. 20.3 shows that the tails of the actual density of $(\bar{x} - \mu)/(s/\sqrt{n})$ are radically different than that of t_{29}. It turns out there is still a huge discrepancy even for $n = 1000$. Unless you get sample sizes in the many thousands, Eq. (20.12) is a very poor description for samples from a standard lognormal density.

What about other population probability densities? Figs. 20.4, 20.5, and 20.6 show plots of the actual $(\bar{x}-\mu)/(s/\sqrt{n})$ density for six different population density functions (some of these we haven't discussed), for sample sizes $n = 5$, 30, and 100, with the t_{n-1} density overlayed.

Examination of these figures makes it pretty obvious that the $n = 30$ guideline isn't always appropriate. If the underlying density is perfectly normal, you can get away with with a sample size of $n = 1$. If the underlying density is very far from normal, you might need n in the *thousands* before you get to single-digit percentage differences in the tails. Sometimes you can greatly reduce the required value of n by first transforming the data (taking a logarithm or something), though it's often not clear ahead of time what type of transformation is required.

What is the take-home message here? *Unless you know the underlying data is described pretty closely by a normal density, any analysis assuming Eq.* (20.12) *could easily be incorrect, unless the sample sizes are truly enormous.*

20.8 The Standard Error of the Mean

It is clear that $s/\sqrt{n}$ in Eq. (20.12) is playing a similar role to $\sigma/\sqrt{n}$ in Eq. (20.4). Since we don't know $\sigma/\sqrt{n}$, we are essentially using $s/\sqrt{n}$ as our best guess

for the standard deviation of the SMD, and using this guess takes us from a normal to a t density. The quantity $s/\sqrt{n}$ is called the `standard error` or `standard error of the mean`, sometimes just *se* or *sem* for short.

$$\frac{s}{\sqrt{n}} = se \text{ or } sem \tag{20.14}$$

We could then characterize Eq. (20.12) as saying that the residual $\bar{x} - \mu$ of a sample's mean, measured in units of the sample's standard error, follows a student's t distribution with $n-1$ degrees of freedom. In the next chapter we will start using this result in a variety of frequentist methods.

20.9 Exercises

Assume that the weight w of coyotes in North Dakota is modeled by a normal density, i.e. $w \sim \text{normal}(\mu, \sigma)$.

1. Do some research to find estimates for μ and σ. Give your sources and justifications. Make a plot of your normal density.

2. In real-life a researcher might randomly select $n = 100$ coyotes and find their weights, thereby obtaining a size $n = 100$ sample. We will now imagine that the researcher obtained 100,000 such samples! Of course that's not possible in real-life, but it's possible in R!

   Use R to simulate drawing $N = 10^5$ samples of size $n = 100$ from the normal density you defined in question 1. Find the sample-mean of each of the N samples, and store these N sample means to a vector. Plot a histogram of these sample means, making sure to set `freq=FALSE` when calling `hist()`. Overlay the lines of the original normal density and also of the SMD predicted by Eqs. (20.2) and (20.3). Which of these two lines best fits the histogram?

3. Repeat the previous question for $n = 4$. How does the resulting histogram change compared to the $n = 4$ case? Also compute and compare the standard error in both cases.

4. A community member is trying to understand why the SMD gets narrower as n gets larger. Write a paragraph or two explaining this phenomenon in clear, simple language.

CHAPTER 21

Confidence Intervals

An alternative to the Bayesian credible interval (let's denote it BCI) is the frequentist confidence interval, which we will denote FCI. The FCI is far more common than the BCI, because in simple statistical tasks the FCI was easier to compute in the absence of modern computers, and through cultural inertia it's still the defacto standard. One downside of the FCI is that it's tougher to interpret as a probability, as discussed in Sec. 21.2. Another downside is that it's difficult to adapt to complex models. For example, even in the simple case of comparing the means of two normal groups with unknown standard deviations, discussed in Sec. 21.6, the underlying density (the *Behrens-Fisher density*) is so complicated that computing confidence intervals is just as involved as the full Bayesian computation.

We introduce the idea of the FCI in the next section in the context of an artificial problem which basically never arises: trying to determine the mean μ for a normal distribution for which we already know the standard deviation σ. After introducing the idea of the FCI in this artificial context, in the following sections we discuss the more realistic situation where neither μ nor σ is known.

21.1 95% FCI With Known σ

To begin, consider drawing samples of size n from a normal population with known μ and σ. Each sample will have its own mean $\bar{x}$ and standard deviation s. Some of the sample means will be pretty far from μ, some will be pretty close, etc.

As an illustration, consider the specific case of a normal population with $\mu = 6$ and $\sigma = 2$. Consider a "trial" to be drawing a random sample of size $n = 5$ from the population, and computing the sample mean $\bar{x}$. According to Eqs. (20.2) and (20.3), the resulting distribution of sample means (after a very large number of trials) should be a normal distribution with mean $\mu = 6$ and standard deviation $se = \sigma/\sqrt{n} = 2/\sqrt{5} \approx 0.89$. This distribution is shown in Fig. 21.1.

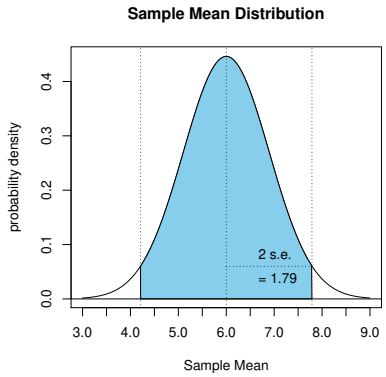


Figure 21.1: Sample mean distribution for $n = 5$, $\mu = 6$, $\sigma = 2$. Notice that about 95% of the sample means obtained will fall within 2σ of μ (shaded region).

We first observe that the interval $(\mu - 2se, \mu + 2se)$, indicated by the blue shaded region, gives an area of about 0.95, as can be verified with R:

(R.21.1)
```
mu <- 6
se <- 2/sqrt(5)
prob <- pnorm(mu+2*se,mean=mu,sd=sigma)
        - pnorm(mu-2*se,mean=mu,sd=sigma)
```

There is thus a 95% probability that a randomly chosen $n = 5$ sample will have a sample mean $\bar{x}$ that falls within the interval $\mu \pm 2se$. Just to be clear, this interval $\mu \pm 2se$ is *not* what is meant by "the 95% confidence interval", which we'll get to in just a moment.

Using R we can simulate drawing 100 $n = 5$ samples from the given population. A

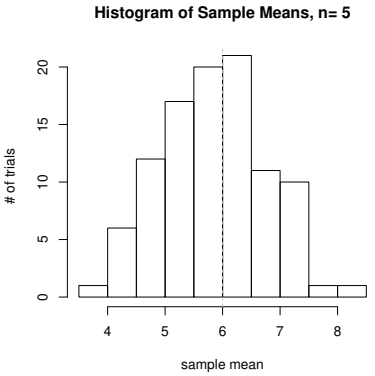


Figure 21.2: Histogram of means for 100 randomly generated $n = 5$ samples from a population with $\mu = 6$ and $\sigma = 2$.

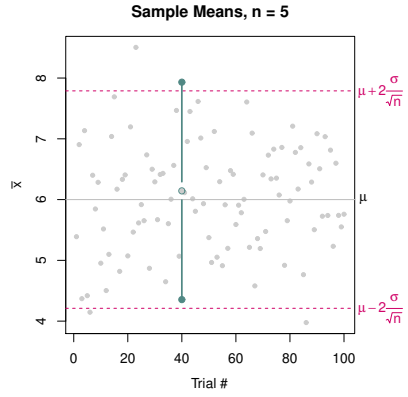


Figure 21.3: Means for 100 randomly generated $n = 5$ samples from a population with $\mu = 6$ and $\sigma = 2$. The horizontal dotted lines are $\mu \pm 2se$, where $se = \sigma/\sqrt{n}$. The green line stretches $2se$ above and below the $\bar{x}$ for random sample #40.

histogram of the results is shown in Fig. 21.2. Another way of visualizing the data is to plot trial # along the horizontal axis, and sample mean for that trial on the vertical axis. The result is shown in Fig. 21.3. Notice that only 3 of the 100 dots fall outside of the red dotted lines. In other words, 97 of our 100 simulated samples had means $\bar{x}$ within $2se = 1.79$ of the true mean $\mu = 6$. If we ran the simulation again, we might get 93 or 95 or even 89. On average, however, 95 of the samples will have means falling within the dotted lines.

We will now look at our simulated data from a slightly different point of view. Let us focus our attention on a randomly selected dot in Fig. 21.3. Notice that if we happened to pick one of the 97 dots falling between the dotted lines, then the value of $\bar{x}$ for this dot will be at most $2se$ above μ (i.e. our value of $\bar{x}$ is greater than μ), or at most $2se$ below μ (i.e. our value of $\bar{x}$ is less than μ). And on average, 95% of sample means will be at most $2se$ above or $2se$ below μ. *This is the case even when we don't know the value of* μ. This fact is very important. It means that if I look at one of the sample means, if it happens to be one of the 95% of "good samples", then if I walk $2se$ in either direction from $\bar{x}$, I should have run into the true mean μ at some point.

To visualize this, look at the dots in Fig. 21.3. If you pick any dot between the two red dotted lines, and go a distance $2\sigma/\sqrt{n} = 1.79$ above and below the dot, you'll somewhere hit the mean-line at $\mu = 6$. This is shown for sample #40 in the figure. The green line segment stretches 1.79 up and down from the value of $\bar{x}$, and indeed it crosses the $\mu = 6$ line. Now imagine that everything on the graph (the lines and all the other dots) aside from your dot gets removed. You would have no idea as to the value of μ. But you know that if you sweep out 1.79 above and below your dot, that at some point you'll have passed over the mean. On the other hand, you don't know ahead-of-time if your dot is actually one of the 95% of "good dots". If you happen to have one of the 5% of "bad dots", then even if you go 1.79 above and below, you wont actually pass over μ.

In other words, on average, 95% of the random samples collected (each with its own $\bar{x}$) will define an interval $(\bar{x} - 2se\ ,\ \bar{x} + 2se)$ that will contain the true population mean μ. On the other hand, if you compute this interval for the other 5% of the samples (the "bad samples"), it will not contain μ. This is the meaning of the 95% confidence interval.

To summarize: consider repeatedly drawing size n samples from a normal population with standard deviation σ. For each of these samples, construct an interval $(\bar{x} - 2\sigma/\sqrt{n}, \bar{x} + 2\sigma/\sqrt{n})$. On average, 95% of these intervals will contain the true population mean μ. The other 5% of these intervals will not. The above interval is known as the `95% confidence interval` for the sample. The green line in Fig. 21.3 is the visualization of the 95% confidence interval for sample #40 in our simulation.

It is important to realize that the 95% confidence interval depends upon the sample mean $\bar{x}$, and hence each sample will have its own confidence interval. As such, a 95% confidence interval is just as random as the sample itself. To get a feeling for this, in Fig.21.4 below we have graphed the 95% confidence intervals for the dots in Fig. 21.3. For clarity the black dots showing $\bar{x}$ have been removed.

Notice that the three sample means $\bar{x}$ outside the red dotted lines in Fig. 21.3 result in 95% confidence intervals that do not include μ. These "bad" confidence intervals are colored in red in Fig. 21.4.

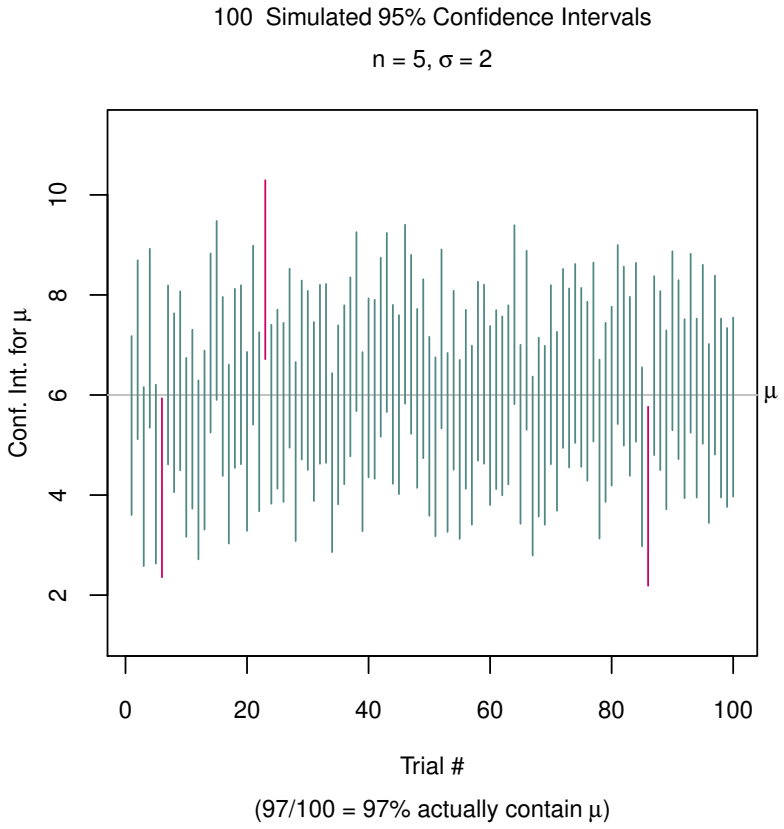


Figure 21.4: 95% Confidence intervals for the 100 samples shown in Fig. 21.3. The three dots outside of the red-dotted lines in Fig. 21.3 denote samples with 95% confidence intervals that do not include the true population mean μ. These three intervals are colored in red.

21.2 Interpretation

How do we interpret the 95% FCI in our artificial situation? Imagine we draw a sample of size n from a normal density with known σ. We compute $\bar{x}$ for our sample, and then construct the interval $\bar{x} \pm 2\sigma/\sqrt{n}$. If we happened to draw one of the 95% of "good samples", this interval will contain μ. We say "we are 95% confident that μ is in our interval". Does this mean we can say "there is a 95% probability that μ is in this interval"? No. In a frequentist analysis, you ascribe probability to the data (which generates our FCI), not to the parameter μ. We regard the value of μ as a fixed number, and hence for any given interval, either μ is in the interval or not. In other words, logically there is either a 0% or a 100% probability that μ is in the FCI.

Think of μ as being a number that's nailed-down to a number line, and that every time we draw samples (data) we randomly generate a new 95% FCI that we throw at the line. 95% of these intervals will cover the nailed-down value of μ, and 5% won't. In other words, the 95% gives the probability that our randomly generated interval (computed from the data) will *contain* or *cover* the true value of μ.

Sometimes the confidence level (e.g. 95%) is called a *coverage probability*.

In contrast, we *would* say there is a 95% probability that μ lies in the 95% Bayesian credible interval, because in a Bayesian analysis we ascribe probability to parame-

ters like μ, and not to the data D that would generate something like a confidence interval.

In fact, it is not uncommon for people to validate a tough-to-interpret frequentist method, like the FCI, by seeing how closely it lines up with the numerical results of an easier-to-interpret Bayesian analysis.

Ok, so the FCI and BCI are technically different, blah blah. Does any of that actually make a difference in terms of the numbers we report or the decisions we make? For simple statistical models (single samples with things like normal densities), FCIs tend to match BCIs computed with non-informative priors. In such cases, it's typically okay to just compute an FCI, since there is less computation involved. On the other hand, if the models become more complicated, or if there is relevant prior information at hand, then typically it's better to do a Bayesian analysis if possible.

21.3 Formally Constructing the FCI

It will be useful to have a more formal mathematical procedure for constructing FCIs. To begin, we will reformulate the calculation in terms of the standard normal density. Let's take another look at Eq. (20.4):

$$\frac{\bar{x}-\mu}{\sigma/\sqrt{n}} = z \tag{21.1}$$

The RHS is a standard normal variable. This means that every time we take a sample, we'll essentially get a different value of z drawn from a standard normal density. What values of z are we likely to get? For example, what even-tailed interval will contain 95% of the z-values? We can use the `qnorm()` function to find the corresponding quantiles:

```
CL <- 0.95
zL <- qnorm(0.5-CL/2)
zR <- qnorm(0.5+CL/2)
```
(R.21.2)

Incidentally, this shows that the values of ± 2 we used earlier were approximate.

giving $z_L = z_{0.025} = -1.96$ and $z_R = z_{0.975} = +1.96$. Thus, 95% of z values drawn from the standard normal density will be in the interval $z_L < z < z_R$, meaning $-1.96 < z < 1.96$ for 95% of the samples drawn. If this is true of the RHS of Eq. (20.4), it is also true of the LHS:

$$z_L < \frac{\bar{x}-\mu}{\sigma/\sqrt{n}} < z_R \tag{21.2}$$

We can also write this as two separate inequalities:

$$z_L < \frac{\bar{x}-\mu}{\sigma/\sqrt{n}} \qquad \text{AND} \qquad \frac{\bar{x}-\mu}{\sigma/\sqrt{n}} < z_R \tag{21.3}$$

Solving the two inequalities for μ yields

$$\mu < \bar{x} - z_L \cdot \frac{\sigma}{\sqrt{n}} \qquad \text{AND} \qquad \mu > \bar{x} - z_R \cdot \frac{\sigma}{\sqrt{n}} \tag{21.4}$$

which can be combined to yield the compound inequality

$$\bar{x} - z_R \cdot \frac{\sigma}{\sqrt{n}} \;<\; \mu \;<\; \bar{x} - z_L \cdot \frac{\sigma}{\sqrt{n}} \tag{21.5}$$

so that the confidence interval for μ is

$$FCI = \left(\bar{x} - z_R \cdot \frac{\sigma}{\sqrt{n}} \;,\; \bar{x} - z_L \cdot \frac{\sigma}{\sqrt{n}} \right) \tag{21.6}$$

Note that the left-bound of the FCI involves the right-bound for z, and vice versa. This is just because the minus sign in front of σ in Eq. (20.4) means that bigger values of μ correspond to smaller values of z, and vice versa. For the case of 95%, we had $z_L = -1.96$ and $z_R = +1.96$, giving

$$\begin{aligned} FCI_{0.95} &= \left(\bar{x} - \underbrace{(1.96)}_{z_R} \cdot \frac{\sigma}{\sqrt{n}} \ , \ \bar{x} - \underbrace{(-1.96)}_{z_L} \cdot \frac{\sigma}{\sqrt{n}} \right) \\ &= \left(\bar{x} - 1.96 \cdot \frac{\sigma}{\sqrt{n}} \ , \ \bar{x} + 1.96 \cdot \frac{\sigma}{\sqrt{n}} \right) \end{aligned} \tag{21.7}$$

For a general confidence level p, Eq. (21.6) still applies if we compute the appropriate z quantiles. For example, to compute a 80% confidence interval, we have that $z_L = z_{0.10}$ and $z_R = z_{0.90}$ so that

$$FCI_{0.90} = \left(\bar{x} - z_{0.90} \cdot \frac{\sigma}{\sqrt{n}} \ , \ \bar{x} - z_{0.10} \cdot \frac{\sigma}{\sqrt{n}} \right) \tag{21.8}$$

and for a confidence level p we have

$$FCI_p = \left(\bar{x} - z_{(1+p)/2} \cdot \frac{\sigma}{\sqrt{n}} \ , \ \bar{x} - z_{(1-p)/2} \cdot \frac{\sigma}{\sqrt{n}} \right) \tag{21.9}$$

When computing quantiles for the even-tailed interval containing probability p, we have normally used $0.5 - p/2$ and $0.5 + p/2$. Algebraically, these are equivalent to $(1-p)/2$ and $(1+p)/2$.

So much for the idealized situation of knowing σ. Now let's return to reality and construct FCIs when we don't know σ.

21.4 FCIs for μ With Unknown σ

When σ is unknown, instead of starting with Eq. (20.4) we start with Eq. (20.12). Repeating the steps of Sec. 21.3, Eq. (21.9) becomes

$$FCI_p \text{ for } \mu = \left(\bar{x} - t_{(1+p)/2} \cdot \frac{s}{\sqrt{n}} \ , \ \bar{x} - t_{(1-p)/2} \cdot \frac{s}{\sqrt{n}} \right) \tag{21.10}$$

where t is a student's t density with $n-1$ degrees of freedom. Note that the width of the FCI now varies from sample to sample, since s varies from sample to sample. The interpretation is still the same, though. If you happened to select one of the fraction p of "good samples", then the FCI computed with Eq. (21.10) will contain the true mean μ. If you got unlucky and picked one of the fraction $1-p$ of "bad samples", then your FCI won't actually contain μ. We can redraw Fig. 21.4 by using the same sample data to construct FCIs based on t. The result is shown in Fig. 21.5. The figure makes it clear that each sample has a different confidence interval, both in location and size.

Happily, rather than having to use Eq. (21.10), we can use the built-in function `t.test()`. This function returns a list with a variety of different elements, one of which is `conf.int`. For example, to compute the 90% confidence interval on a vector of sample data `x`, we issue the commands

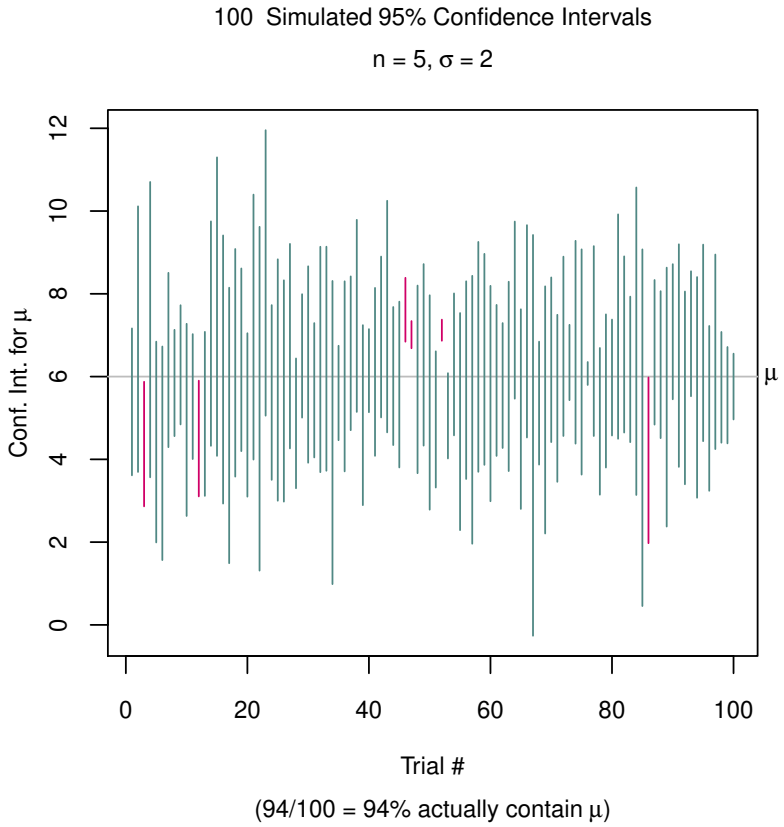


Figure 21.5: 95% t-based Confidence intervals for the 100 samples shown in Fig. 21.3. The intervals colored in magenta denote sample FCIs that don't include the true mean μ. Note that these "bad intervals" are different from those of Fig. 21.4, even though the sample data is the same. This is simply because the FCIs computed with the t-statistic are different than those computed with the z-statistic.

(R.21.3)
```
t.test(x,conf.level=0.90)$conf.int
```

We'll see the `t.test()` function again when we discuss null hypothesis significance testing (NHST).

Example 21.1: Compute FCI For Crab Data

Problem: Compute the 95% frequentist confidence interval for the crab temperature data of Sec. 1.3, and compare this with the 95% Bayesian credible interval computed with a uniform (flat) prior.

Solution: Assume the temperatures are drawn from a population normal density of mean μ and standard deviation σ. Because σ is unknown, we must construct the confidence interval using the t distribution, as in Eq. 21.10:

```
x <- c(25.8, 24.6, 26.1, 22.9, 25.1, 27.3, 24.0, 24.5,
       23.9, 26.2, 24.3, 24.6, 23.3, 25.5, 28.1, 24.8,
       23.5, 26.3, 25.4, 25.5, 23.9, 27.0, 24.8, 22.9,
       25.4 )
FCI <- t.test(x,conf.level=0.95)$conf.int
```

The default value of the `conf.level` parameter is 0.95, so we could also type `t.test(x)$conf.int` here.

which gives the 95% FCI as $(24.5, 25.6)$°C. There a 95% probability our interval is one of the "good ones" that contains the true value of μ.

Note that the FCI matches the BCI obtained in Sec. 18.4. As we indicated previously, for simple models such as this one, the FCI tends to agree with the BCI resulting from an uninformed prior.

21.5 FCI for σ

Typically people only focus on the FCI for μ. However, there are situations in which you would want the FCI for σ. There isn't a built-in function for this in base R, so we'll need to compute it ourselves. First recall Eq. (20.10):

$$\frac{(n-1)s^2}{\sigma^2} = \chi^2_{n-1} \qquad \text{where} \qquad \chi^2_{n-1} \sim \text{chisq(n-1)} \tag{21.11}$$

For a confidence level p, we first find the numbers χ^2_L and χ^2_R such that a fraction p of the time somebody draws a sample they'll obtain a value of χ^2_{n-1} between χ^2_L and χ^2_R. One way to do this is to get quantiles at $(1-p)/2$ and $(1+p)/2$. For example, for the crab data of Ex. 21.1, assuming the code from Ex. 21.1 has been run, we can compute the quantiles for a 95% interval as

(R.21.4)
```
p <- 0.95     # confidence level
pL <- (1-p)/2
pR <- (1+p)/2
n <- length(x)
chisqL <- qchisq(pL,df=n-1)
chisqR <- qchisq(pR,df=n-1)
```

If the RHS of Eq. (21.11) falls between χ^2_L and χ^2_R for a fraction p of samples, then so does the LHS:

$$\chi^2_L < \frac{(n-1)s^2}{\sigma^2} < \chi^2_R \tag{21.12}$$

This can be written as two separate inequalities:

$$\chi^2_L < \frac{(n-1)s^2}{\sigma^2} \qquad \text{AND} \qquad \frac{(n-1)s^2}{\sigma^2} < \chi^2_R \tag{21.13}$$

In solving these inequalities, note that σ and χ^2 and s are all positive numbers by definition, so that there's no worrying about flipping the sign of the inequality as we're dividing/multiplying, and there is no issue in taking square-roots.

We can now solve these two inequalities for σ. We obtain

$$\sigma < s\sqrt{\frac{n-1}{\chi^2_L}} \qquad \text{AND} \qquad \sigma > s\sqrt{\frac{n-1}{\chi^2_R}} \tag{21.14}$$

Thus, for a confidence level p, we have that the FCI for σ is

$$FCI_p = \left(s\sqrt{\frac{n-1}{\chi^2_R}} \ , \ s\sqrt{\frac{n-1}{\chi^2_L}} \right) \tag{21.15}$$

or

$$FCI_p = \left(s\sqrt{\frac{n-1}{\chi^2_{(1+p)/2}}} \ , \ s\sqrt{\frac{n-1}{\chi^2_{(1-p)/2}}} \right) \tag{21.16}$$

where the χ^2 are quantiles of a chi-squared density with $n-1$ degrees of freedom. Note that the lower bound on the FCI corresponds to the upper bound of χ^2, which

results from the fact that σ is in the *denominator* of Eq. (21.11). Continuing with the crab data, we compute the left and right bounds of the FCI for σ via

(R.21.5)
```
s <- sd(x)   # compute sample standard deviation
sigma.FCI.L <- s*sqrt((n-1)/chisqR)
sigma.FCI.R <- s*sqrt((n-1)/chisqL)
```

which gives the following 95% FCI for σ:

$$FCI_{0.95} = (1.05, 1.87)^\circ \text{ C} \tag{21.17}$$

Note that this agrees closely with both the ETI and HDI Bayesian credible intervals computed in Ch. 18. As we stated in Sec. 21.4, in simple situations like this the FCI will match the BCI computed with non-informative priors.

21.6 Group Comparisons Are Problematic

For a single sample of data drawn from a normal density, we have seen that the FCI and BCI agree very closely. It is then natural to wonder how the frequentist analysis compares to the Bayesian analysis when comparing two groups of data from two normal densities.

Consider two samples of data, A and B, with sample means $\bar{x}_A$ and $\bar{x}_B$, and with sample standard deviations s_A and s_B. Assume the data are modeled as

$$x_A \sim \text{normal}\,(\mu_A, \sigma_A)$$
$$x_B \sim \text{normal}\,(\mu_B, \sigma_B)$$

If we compute the FCIs for μ_A and μ_B separately, they will agree with the BCI for these parameters. However, remember that when comparing groups we want to consider the parameter $d\mu \equiv \mu_A - \mu_B$, i.e. we want to examine the *difference* in the two population means. Once we have draws for μ_A and μ_B in our Bayesian analysis, we simply subtract them to get draws for $d\mu$. It was trivial to compute various probabilities related to $d\mu$. However, in a frequentist analysis, there is no simple way to deal with $d\mu$.

The problem is that while the individual sample means $\bar{x}_A$ and $\bar{x}_B$ are distributed according to Eq. (20.12), there is no simple formula for distribution in the difference of two sample means. In fact, the actual distribution of $d\mu \equiv \mu_A - \mu_B$ is called the *Behrens-Fisher* distribution, and there is no simple formula for it. You can compute it *approximately* using computers, but at this point things are just as complicated as computing the Bayesian posterior. Given all this, there is no easy way to use FCIs for the individual μ values to construct an FCI for $d\mu$.

At this point, one of two things is normally done in a frequentist analysis:

1. Assume that $d\mu$ is actually described by a t distribution
2. Assume that $d\mu$ is described by a modified t distribution known as Welch's t distribution, which tends to give better results if σ_A and σ_B are substantially different.

We will examine both of these approaches in more detail when we consider NHST. Unfortunately, both of these approaches are incorrect, because $d\mu$ is not described

by any type of t distribution. Furthermore, without knowing the population standard deviations ahead of time, it's hard to know how much of an error results from using a t distribution instead of the correct Behrens-Fisher distribution.

So if it's wrong, why is $d\mu$ so commonly modeled as a t distribution in frequentist analyses? Because doing the correct analysis was hard in the pre-computer era, and even though we have computers now, the popular "wisdom" of using a t distribution is still around.

The take home is: if you have to compare groups, you are *much* better off using a Bayesian analysis.

21.7 Exercises

1. How does the frequentist view of probability differ from the Bayesian view?
2. What is the meaning of the FCI? How does it compare to the BCI?
3. Let's revisit the Western Meadowlark data from the Ch. 18 Exercises.
    i) Compute the 50, 80, 95, and 99% FCI for the population mean μ and population standard deviation σ.
    ii) What assumptions did you make in computing these FCIs?
    iii) How do these intervals compare to the corresponding BCI you computed in the Ch. 19 Exercises?
4. Consider the corn height data from Ch. 19.
    i) Compute 95% FCI for μ_A, μ_B, and μ_C.
    ii) How do these FCI compare to the 95% BCIs?
    iii) Explain the difficulty in constructing an FCI for quantities such as $d\mu_{AB} \equiv \mu_A - \mu_B$.

CHAPTER 22

NHST I - Basic Methodology

The method of null hypothesis significance testing, or NHST, is extremely prevalent. Unfortunately, as many researchers and organizations (including the American Statistical Association [10]) have pointed out, the method is actually rather poor in the context of scientific research. So why are we discussing it here? There are three reasons. One, you might be expected to perform NHST at some point. Two, so you can better understand other statistics books and scientific papers, which heavily employ these methods. And three, so you can be appropriately critical of claims that are made on the basis of these methods.

22.1 Motivating The Logical Framework

A typical problem in NHST is the following. There are two groups of plants. Group A is treated with a fertilizer, and group B is untreated. We want to measure plant heights and see if the data D indicates that the fertilizer helps plant growth. If we imagine the height data h_A are drawn from a normal density with mean μ_A, and likewise the h_B are drawn from a normal density with mean μ_B, we would be interested in knowing if $\mu_A > \mu_B$. If yes, then presumably the fertilizer helps.

In a Bayesian analysis, we could define the parameter $d\mu \equiv \mu_A - \mu_B$, and then ask the question "What is $d\mu$?" The answers are different intervals of $d\mu$ and the corresponding probability, i.e. the posterior density for $d\mu$. We could also compute things such as $\mathbb{P}(d\mu > x)$, i.e. the probability that μ_A is at least x bigger than μ_B.

In the frequentist framework, however, $d\mu$ is not the direct outcome of some repeatable event, and hence you can't ascribe it a probability. Instead, we must assign probability to the data. One way of doing this is to compute a confidence interval. However, as discussed in Sec. 21.6, it is rather difficult to compute FCIs for something like $d\mu$. And even if you *did* use the Behrens-Fisher distribution to compute an FCI for $d\mu$, it still doesn't make sense to ask "what is the probability that $\mu_A > \mu_B$?" So how do you decide if $\mu_A > \mu_B$ in the frequentist framework?

The approach of NHST is to first hypothesize that the means are equal, $\mu_A = \mu_B$. This is equivalent to hypothesizing that $\mu_A - \mu_B = 0$. Since *null* is another word for zero, this hypothesis of zero difference or *null* difference is known as a `null hypothesis`, and is often denoted H_0:

$$H_0: \quad \mu_A = \mu_B \quad \Longleftrightarrow \quad d\mu = 0 \tag{22.1}$$

Again, in frequentist methods we can't assign probabilities to H_0. Instead, the goal of NHST is to see if we can *reject* H_0 entirely by showing that the data D are sufficiently improbable if H_0 is assumed to be true.

Before dealing with the comparison of two samples like A and B, we'll illustrate

some of the ideas in the simpler context of comparing a single sample against a known population mean.

22.2 Single–Sample t tests

Assume that you've been growing unfertilized tomato plants in your greenhouse for years, and that on average they grow to a mean height of 42 in. Recently you tried out some fertilizer on 4 of your tomato plants, and found they attained heights of 54.3, 41.1, 49.9, and 55.1 inches. The question is: did the fertilizer help?

22.2.1 Compute p Value

Assume that the four fertilized plants had measured heights drawn from a normal density of mean μ and standard deviation σ. We now establish a null hypothesis that the fertilized plants don't grow any taller than unfertilized plants:

$$H_0: \quad \mu = 42 \quad \Longleftrightarrow \quad \mu - 42 = 0 \tag{22.2}$$

Incidentally, because we suspect fertilizer could help the plant growth, we can keep in mind an alternative hypothesis H_A that $\mu > 42$.

$$H_A: \quad \mu > 42 \quad \Longleftrightarrow \quad \mu - 42 > 0 \tag{22.3}$$

For now we are choosing to exclude the possibility that fertilizer hurts plant growth, i.e we are just assuming there is zero probability of $\mu < 42$.

Unfortunately, NHST is difficult to use directly with a compound hypothesis like H_A, so we will focus on H_0.

Our 4 sample heights have a mean of $\bar{h} = 50.1$ in and standard deviation $s = 6.5$ in. Note that $\bar{h}$ is a lot bigger than 42 inches. We now ask our key question:

> Assuming H_0 were true, how probable would it be to get a sample mean at least as big as 50.1 inches (assuming s is approximately 6.5)?

Recall that the sample mean distribution (SMD) for samples of size n drawn from a normal density of mean μ obeys Eq. (20.12):

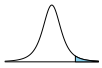
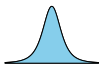
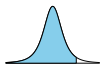
$$\frac{\bar{h} - \mu}{s/\sqrt{n}} = t_{n-1} \tag{22.4}$$

where $n - 1 = 4 - 1 = 3$ is the degrees of freedom of the t distribution. If we assume H_0 is true, then $\mu = 42$. Plugging in the $\bar{h}$, s, and n for the sample, we have that

$$t_3 = \frac{50.1 - 42}{6.48/\sqrt{4}} = 2.523. \tag{22.5}$$

Note that if we had obtained a value of $\bar{h}$ higher than 50.1, then t_3 would also be higher. Thus, the probability of getting data as "extreme" as what we got is the

same as the probability of drawing a value $t > 2.523$ from a t density with $n = 3$ degrees of freedom. This is easy to compute in R using the cumulative probability function `pt()`:

$$
\begin{aligned}
\mathbb{P}(t \geq 2.523) &= \text{area between } 2.523 \text{ and } \infty \\
&= \quad \text{  } \\
&= \Big(\text{area between } -\infty \text{ and } +\infty\Big) \\
&\qquad - \Big(\text{area between } -\infty \text{ and } 2.523\Big) \\
&= \quad \text{  } \quad - \quad \text{  } \\
&= 1 - \texttt{pt(2.523,df=3)} \\
&= 1 - 0.957 \\
&= \boxed{0.043}
\end{aligned}
\tag{22.6}
$$

The resulting probability of $p = 0.043$ denotes the probability of obtaining data at least as extreme as what we got assuming H_0, and is referred to as a `p-value`. To get a p-value that is smaller than 0.043, a sample would need to have an $\bar{h}$ larger than 50.4 or an s smaller than 6.48.

22.2.2 Significance Levels

Let's think about our p-value in frequentist terms. Imagine that H_0 is true, i.e. that $\mu = 42$, and a huge number of different people took size $n = 4$ samples from the population distribution. Each person computes $t = (\bar{h} - \mu)/(s/\sqrt{n})$ for their sample. Only 4.3% of the people (i.e. of the samples) will have a t value greater than our value of 2.523. This means 95.7% of people will have a t value less than what we got.

Here are two possible explanations for our high value of t, and correspondingly low value of p:

A) H_0 is true, but we were one of the unlucky 4.3% of people who got a t this big.

B) H_A is true, hence our data isn't as unlikely as the p-value would suggest.

There is actually another explanation: our data was unlikely under both H_0 and H_A. Unlikely things happen all the time. The failure to acknowledge this 3rd explanation is a common source of misunderstanding in the NHST method, as we will discuss in the next chapter.

So which is the correct explanation? We can't know for sure, but note that as the p-value gets smaller, explanation A seems less and less plausible. For example, if in a different problem we computed $p = 0.000001$, then B seems more likely than us having 1-in-a-million bad luck with our sample.

The convention in NHST is the following: before doing the analysis, choose a cut-off value for p called the `significance level`, denoted α. If our p-value is less than α, then we'll say explanation A is too unlikely, and hence we'll assume explanation B is correct. In other words, if $p < \alpha$, then we would reject the null-hypothesis H_0. On the other hand, if $p > \alpha$, then explanation A isn't too unlikely, and we therefore fail to reject H_0.

Rejecting H_0 on the basis of the p-value involves subtle logical issues that we address in more detail in the next chapter.

The most common choice for the significance level α is 5%, but it is ultimately arbitrary and you can pick whatever value you like. The conventional wisdom is

to pick α before doing the calculation. Assume we had chosen the standard value of $\alpha = 0.05$. In our problem, therefore, we have $p = 0.043$ and hence $p < \alpha$. Thus, our final result would be to reject H_0, i.e. we reject the hypothesis that $\mu = 42$. This means we are claiming that the fertilizer has a non-zero impact on the growth of the tomato plants.

22.2.3 One-Tailed vs. Two-Tailed Tests

Our p value of 0.043 was the probability that $\bar{h}$ would be at least 8.1 units *greater* than μ (or that s would be smaller than 6.48), which aligns with the alternative hypothesis Eq. (22.3) we had in mind. Because we computed the probability contained in one tail of the t-density, we conducted a `one-tailed test`. If you have reason to believe that this substance you're adding would hurt plant growth, you would compute a one-tailed test on the left-hand tail.

On the other hand, what if we weren't sure whether this substance we're adding to the tomato plants would help or hurt growth? Then our alternative hypothesis would be that either $\mu > 42$ or $\mu < 42$. In this case, we would want to compute the probability that $\bar{h}$ was at least 8.1 units away from 42 in *either direction*, meaning we would have added $\mathbb{P}(\bar{h} > 50.1)$ and $\mathbb{P}(\bar{h} < 33.9)$, which would have essentially doubled the probability to $p = 0.086$. This would have been a `two-tailed test`.

Since we have no reason to suspect that the fertilizer would *hurt* plant growth, considering the $\mu < 42$ possibility seems unneccesary here, and we'll stick with the one-tailed test.

Example 22.1: Single-Sample t-Test

Why aren't the measurements all the same? When you weigh objects on a scale, sometimes you get slightly different numbers due to small amounts of noise: maybe some extra friction in the scale mechanism given the way the mass is distributed over the plate, or maybe there were small electrical fluctuations in the circuitry, etc. All measurements of any type are noisy to some extent.

Problem: Before weighing soil samples you want to check that a scale is calibrated accurately. To do so, you take a calibration test object that is known to have a mass of 500.0 grams, and you perform 50 measurements on the scale. You find that the 50 readings have a mean of 500.4 grams, with standard deviation 2 grams.

How probable are these measurements if the scale is perfectly calibrated? Using a cutoff of $\alpha = 5\%$, do we decide that the scale is sufficiently calibrated?

Solution: If we assume the scale is perfectly calibrated, we can think of the 50 mass measurements m_i as draws from a normal density with mean $\mu = 500.0$ g and some small standard deviation σ characterizing the tiny fluctuations we get from measurement to measurement (due to noise).

Our null hypothesis is then $H_0 : \mu = 500.0$. We don't know ahead of time whether the scale is calibrated too heavy or too light, so we perform a two-tailed test. There are $n = 50$ measurements, so the t distribution has $n - 1 = 49$ degrees of freedom. Using Eq. (20.12) we have

$$t_{49} = \frac{\bar{m} - \mu}{s/\sqrt{n}} = \frac{500.4 - 500.0}{2/\sqrt{50}} = 1.41 \tag{22.7}$$

The two-tailed p-value is the probability of getting data this far from $t = 0$

in either direction. Hence, we wish to compute

$$\begin{aligned}\mathbb{P}(t < -1.41 \text{ or } t > 1.41) &= \\ &= 2 \times \\ &= \texttt{2 * pt(-1.41,df=49)} \\ &= \boxed{0.16} \end{aligned} \tag{22.8}$$

We can simply double the area in one tail here, because the t distribution is symmetric around $t = 0$. Keep in mind that many distributions, such as χ^2 and F, are not symmetric.

There is thus a 16% probability of getting data at least as far-off as you got, i.e. $p = 0.16$. Since p is not smaller than $\alpha = 0.05$, you would fail to reject the null hypothesis H_0 that the scale is perfectly calibrated.

Important point: Failing to reject H_0 does not mean that you *accept* H_0. Thus, in a logical sense, we haven't really verified that the scale is calibrated. We simply have no reason to suspect it *isn't* calibrated.

If this logical point confuses you, consider the following: Let H_0 be the hypothesis that bigfoot exists. While you could imagine it is possible to prove that bigfoot exists (say, by capturing one), it is almost impossible to prove that bigfoot *doesn't* exist. If you fail to reject H_0, does that mean you automatically accept H_0?

22.2.4 The `t.test` Function

If we have access to the individual data points, then instead of calculating the t value in Eq. (22.5) manually, and then using `pt()` to compute the p-value of Eq. (22.6), we could instead use the built-in function `t.test()`, which we first saw back in Sec. 21.4.

To compute a one-tailed test with the alternative hypothesis $H_A : \mu > 42$ in mind, we issue the following commands:

(R.22.1)
```
h <- c(54.3,41.1,49.9,55.1)
t.test(h,mu=42,alternative="greater")
```

This gives us the p-value of 0.043 we computed above. The output of `t.test()` is a list containing a variety of information. To only get the p-value, you can issue the code

(R.22.2)
```
p <- t.test(h,mu=42,alternative="greater")$p.value
```

To perform a one-tailed test with the alternative hypothesis $H_A : \mu < 42$ in mind, change the `alternative` parameter to `"less"`. To perform a two-tailed test, just omit the `alternative` parameter (which defaults to `"two.sided"`).

Example 22.2: A Single Sample `t.test()`

Problem: The e-coli levels in a local river had been monitored over the last 30 years, and on average there are 142 colonies/100 mL. A new factory opened near the river last year. The monthly measurements of e-coli levels since the factory opened are shown in Tab. 22.1. Use NHST to determine if the factory has had an impact on the e-coli levels.

month	e-coli
1	153
2	148
3	153
4	212
5	179
6	181
7	158
8	178
9	158
10	177
11	135
12	110

Table 22.1: Monthly e-coli levels (colonies/100 mL) since factory opened.

Solution: We first imagine that the recent monthly measurements are draws from a normal population with mean μ. If the factory has zero (null) impact on the e-coli levels, then we would have $\mu = 142$, which is our null hypothesis H_0. We are not sure if possible pollutants from the factory would increase or decrease e-coli levels, so our alternative hypothesis H_A is either that $\mu < 142$ or $\mu > 142$, i.e. $\mu \neq 142$.

$$\begin{aligned} H_0: &\quad \mu = 142 \\ H_A: &\quad \mu \neq 142 \end{aligned}$$

We are thus interested in a two-tailed test. Unlike Ex. 22.1, here we have the original data points, so we can use `t.test()`. Lets choose the traditional cut-off of $\alpha = 0.05$.

```
ecoli <- c(153,148,153,212,179,181,158,178,158,177,135,110)
p <- t.test(ecoli,mu=142)$p.value
```

which gives a $p = 0.023$. Since $p < \alpha$, we reject H_0, and thus we accept H_A. We decide that the factory has impacted e-coli levels.

22.3 Two-Sample t Tests

In the previous section we tested the mean of a single sample against a known population mean. Let's now consider comparing two samples A and B. Assume the sample data are modeled as

$$\begin{aligned} x_{A,i} &\sim \text{normal}\,(\mu_A, \sigma_A) \\ x_{B,i} &\sim \text{normal}\,(\mu_B, \sigma_B) \end{aligned}$$

meaning they are draws from normal distributions with potentially different means and standard deviations. If the two samples are actually all drawn from the same population, then we would expect $\mu_A = \mu_B$, or equivalently that $\mu_A - \mu_B = 0$. This will be our null hypothesis:

$$H_0: \quad \mu_A - \mu_B = 0 \tag{22.9}$$
$$H_A: \quad \mu_A - \mu_B \neq 0 \tag{22.10}$$

Unfortunately $\mu_A - \mu_B$ is not described by a t distribution, but by the more complicated Behrens-Fisher distribution (as mentioned in Sec.21.6), which doesn't have a simple formula like a t or *normal* distribution. Typically people use an *approximation* to the Behrens-Fisher distribution that was derived by Welch in the 1940s [11], known as `Welch's unequal variances t-test`.

The derivation of the approximation is complicated, so here we just give the result: an approximate t is computed from the data according to

$$t = \frac{\bar{x}_A - \bar{x}_B}{\sqrt{\dfrac{s_A^2}{n_A} + \dfrac{s_B^2}{n_B}}} \tag{22.11}$$

where the quantity t follows a student's t density with degrees of freedom ν com-

puted as

$$\nu = \frac{\left(\frac{s_A^2}{n_A} + \frac{s_B^2}{n_B}\right)^2}{\frac{s_A^4}{n_A^2(n_A-1)} + \frac{s_B^4}{n_B^2(n_B-1)}} \tag{22.12}$$

We could then use `pt()` as usual. Luckily, this complicated formula is automatically used by `t.test()` if you specify two vectors of data.

Example 22.3: A Two-Sample t-Test

Problem: As a variation on the example discussed in Sec. 22.2, imagine that somebody had 13 otherwise equivalent tomato seedlings. Four of them, denoted group A, were regularly treated with a home-brewed fertilizer, and the other 9 (denoted group B) were untreated. The heights of the plants were measured after two months. Did the fertilizer work?

Solution: We can easily compute the sample means as $\bar{h}_A = 50.1$ and $\bar{h}_B = 41.9$. However, as usual, this difference could be due to statistical noise. Assume the heights in group A were draws from a normal population with mean μ_A, and those in group B from a normal population with mean μ_B. Our hypotheses are

$$\begin{aligned} H_0: &\quad \mu_A = \mu_B \\ H_A: &\quad \mu_A \neq \mu_B \end{aligned}$$

In this case we aren't sure if the home-brewed fertilizer would *help* or *hinder* plant growth, so we'll use a two-tailed test. We can use `t.test()` to perform the Welch's unequal-variances t-test. Let's choose $\alpha = 0.05$.

```
hA <- c(54.3,41.1,49.9,55.1)
hB <- c(43.6,46.8,33.9,51.5,33.4,31.5,48.2,40.3,48.1)
p <- t.test(hA,hB)$p.value
```

which gives $p = 0.085$. Since $p > 0.05$, we fail to reject H_0. According to our hypothesis test, the difference in sample means is plausibly attributed to statistical noise even if $\mu_A = \mu_B$, and we thus have inconclusive evidence that the fertilizer affects plant growth.

It's worth noting that had we assumed the home-brew fertilizer was incapable of hindering plant growth, we could have performed a one-tailed test. This gives $p = 0.085/2 < \alpha$, meaning we would reject H_0, meaning we would conclude that the fertilizer *does* improve plant growth.

h_A	h_B
54.3	43.6
41.1	46.8
49.9	33.9
55.1	51.5
-	33.4
-	31.5
-	48.2
-	40.3
-	48.1

Table 22.2: Height data for 4 fertilized plants in group A and 9 unfertilized plants in group B.

This example was adapted from a problem discussed by Jaynes [12]. He notes that if you performed a Bayesian analysis of this problem, you would conclude that $\mathbb{P}(\mu_A > \mu_B) = 0.90$. Even though there is a 90% probability that the fertilized plants grow larger, the NHST method claims the evidence is inconclusive. The Bayesian analysis seems more reasonable here.

22.3.1 Paired t Tests

A variation of the two-sample t test is to consider two samples comprised of measurements of the same collection of entities or objects. Typically this takes the form of *before and after* measurements of something. For example, to see if a pesticide is effective, we could set aside some number n of plants, and count the bugs on each plant before and after the application of the pesticide.

Sometimes a non-paired two-sample t-test is called an `independent t-test` in order to distinguish it from a `paired t-test`.

Let x_i and y_i be the data from each sample. Then x_1 and y_1 would represent the before and after measurements for the first object, etc. We'll define $d_i \equiv x_i - y_i$ as the difference between the two measurements. Assume that the d_i are drawn from a normal density with mean μ. If $\mu = 0$, then there is no difference between the before and after groups. This would be our null hypothesis: $H_0 : \mu = 0$. We then run a single-sample t-test on the data d_i using $\mu = 0$. Alternatively, we could include both of the vectors x and y in `t.test()` and set the `paired` parameter to `TRUE`.

before	after
12	9
12	12
11	12
12	8
13	10
9	9
13	9
13	13
13	11
11	11

Table 22.3: Pb counts (ppb) before and after treatment of water samples with reverse osmosis filter prototype.

Example 22.4: A Paired t Test

Problem: You are trying out a new prototype design for a reverse-osmosis filter that removes lead from water. To test the prototype, you take 10 water samples from the local river (where you know Pb levels are above the safe threshold for drinking). You measure the Pb counts (in parts per billion) of each sample. You then use the prototype filter on the samples, and measure the Pb levels afteward, with results shown in Tab. 22.3. Does the filter reduce Pb levels?

Solution: We can easily check that the mean Pb level before treatment was 11.9, and after treatment was 10.4. This suggests the filter reduces Pb, but of course this could just be statistical noise. Assume the Pb counts are drawn from two different normal populations, with means μ_b and μ_a (before and after). If the filter has no impact on Pb levels, then $\mu_a = \mu_b$, which is our null hypothesis. We expect that the filter will reduce Pb counts, so that $\mu_a < \mu_b$, which is our alternate hypothesis:

$$\begin{aligned} H_0 : \quad \mu_a = \mu_b \quad &\iff \quad \mu_a - \mu_b = 0 \\ H_A : \quad \mu_a < \mu_b \quad &\iff \quad \mu_a - \mu_b < 0 \end{aligned}$$

Let's perform the NHST with $\alpha = 0.05$. First we store the data into two vectors:

```
Pb.b <- c(12,12,11,12,13,9,13,13,13,11)
Pb.a <- c(9,12,12,8,10,9,9,13,11,11)
```

To perform a paired t-test, we can use `t.test()` on the difference of the vectors, setting `mu` to zero:

```
p <- t.test(Pb.a-Pb.b,mu=0,alternative="less")$p.value
```

The order of the vectors is important here. Since we wrote our alternative hypothesis in the form "μ_a is less than μ_b", we are setting the `alternative` parameter to `"less"`, and we must therefore have `Pb.a` listed first and `Pb.b` listed second in `t.test()`. In a two-tailed test the order doesn't matter.

Alternatively:

```
p <- t.test(Pb.a,Pb.b,
            paired=TRUE,alternative="less")$p.value
```

We obtain $p = 0.017$. Since $p < \alpha$, we reject H_0 and accept H_A. We conclude that the filter reduced Pb levels by an amount that is unlikely to be explained by statistical noise.

Keep in mind that the question was only "did the filter reduce Pb" and not "did the filter reduce Pb as much as it should". Google suggests that a good reverse osmosis filter should reduce Pb by 99%. Our prototype may be on the right track, but it still needs a lot of work!

22.4 F-Tests For Group Standard Deviations

Not every hypothesis test uses a t-distribution. For example, again consider two samples of data A and B assumed to be drawn from two different normal populations. If we assume these two populations have the same standard deviation σ, i.e. that $\sigma_A = \sigma_B$, then a bit of algebra shows that the ratio of sample variances is described by

$$\frac{s_A^2}{s_B^2} \sim F_{k,m} \qquad k = n_A - 1, m = n_B - 1 \tag{22.13}$$

where F is modeled by the F density of Eq. (14.9) with $k = n_A - 1$ and $m = n_B - 1$ degrees of freedom. Because the F distribution is not symmetric, the details of computing a p-value are a bit cumbersome for a two-tailed test (see Comment. 22.1 below). Thus, we'll just have R do it for us via the `var.test()` function.

Example 22.5: F-Test For Standard Deviations

Problem: For the tomato data of Ex. 22.3, use NHST with $\alpha = 0.05$ to see if the population standard deviations are different.

Solution: Assume the height data are drawn from normal populations with standard deviations σ_A and σ_B. Our null hypothesis is $H_0 : \sigma_A = \sigma_B$, and our alternative (two-tailed) hypothesis is $H_A : \sigma_A \neq \sigma_B$. We then calculate a p-value via

```
hA <- c(54.3,41.1,49.9,55.1)
hB <- c(43.6,46.8,33.9,51.5,33.4,31.5,48.2,40.3,48.1)
p <- var.test(hA,hB)
```

which gives $p = 0.89$. Since $p > \alpha$, we fail to reject H_0. Our NHST says there is insufficient evidence to claim that σ_A and σ_B are different. However, failing to reject H_0 is logically *not* the same thing as accepting H_0. So we are not claiming that $\sigma_A = \sigma_B$. We are only claiming we that don't have good evidence to the contrary.

We emphasize again that *failing to reject* isn't the same as *accepting*. If I claim that I once made 4,284 free throws in a row, you may not be able to disprove my claim, but that doesn't mean you accept it!

Comment 22.1 – Two-Tailed F Test (Optional Reading)

To understand what `var.test()` is doing in Ex. 22.5, we first compute

$$\frac{s_A^2}{s_B^2} = 0.74 = F_{3,8}$$

Note that it is arbitrary which group we put on top and which group we put on the bottom of Eq. (22.13) (i.e. it is arbitrary which of the groups we labeled A). Hence, we also could have computed

$$\frac{s_B^2}{s_A^2} = \frac{1}{0.74} = 1.35 = F_{8,3}$$

Notice the degrees-of-freedom of the F distribution have flipped.

Our two-tailed test computes the probability of getting F values "at least this

extreme", meaning we would then compute

$$\begin{aligned} p &= \mathbb{P}\big(F_{3,8} < 0.74 \text{ or } F_{8,3} > 1.35\big) \\ &= \mathbb{P}\big(F_{3,8} < 0.74\big) + \mathbb{P}\big(F_{8,3} > 1.35\big) \\ &= \quad + \\ &= \texttt{pf(0.74,3,8)} + \texttt{1-pf(1.35,8,3)} \\ &= 0.89 \end{aligned}$$

which agrees with `var.test()`. A one-tailed test is conceptually simpler, but usually you don't have any prior belief about which of σ_A and σ_B should be larger, meaning you'd want to do a two-tailed test.

22.5 Exercises

1. How do you decide whether to use a single-sample t-test, a Welch's t-test, or a paired t-test? Give an applicable example for each test.

2. Consider the corn data from Tab. 19.1 in Ch. 19.

   i) Perform Welch's t-tests separately on $\mu_A - \mu_B$, $\mu_B - \mu_C$, and $\mu_A - \mu_C$. For each group comparison of population means, indicate whether there is a statistically significant difference between them at a level $\alpha = 0.05$.

   ii) Repeat i) but with F-tests on the population variances.

   iii) How do your results compare with the Bayesian analysis?

3. Consider the Pb data from the exercises in Sec. 19.9. For each pair of locations (i.e. a-b, a-c, etc.) perform an $\alpha = 0.05$ hypothesis test to determine if there is a statistically significant difference in the Pb levels. How do your conclusions compare with those of the Bayesian analysis?

4. Use `install.packages()` to install the `EnvStats` package, and then load it with `library()`. This package includes a dataset `ACE.13.TCE.df` containing measurements of trichloroethylene (TCE) at 10 groundwater monitoring wells before and after remediation. Conduct an appropriate hypothesis test to determine if the remediation was effective in reducing the TCE level.

5. The `EPA.09.Ex.16.1.sulfate.df` dataset from the `EnvStats` package contains sulfate measurements at a background well and a downgradient well. Conduct an appropriate hypothesis test to determine if the two wells have different sulfate levels.

The `EnvStats` package loads its own version of `t.test()` which formats the output differently. If you want the default version from base R, then you can store the data to a custom variable name and then issue the command `detach("package:EnvStats",unload=TRUE)` to unload the package.

CHAPTER 23

NHST II - Pitfalls

Null hypothesis significance testing is very widely used, but in reality it is typically a poor method of statistical inference in scientific situations. In the previous chapter we presented a number of example NHST calculations without spending too much time on issues of interpretation. In actuality, all of these examples contained many logical deficiencies that are inherent to the NHST process. Before moving on to further aspects of NHST, in this chapter we discuss the various pitfalls, logical shortcomings, and typical misuses of NHST.

23.1 Issues With Interpreting p Values

Consider again the example of Sec. 22.2, where we wanted to see if a fertilizer resulted in improved growth from the usual value of 42 inches. We first established hypotheses:

$$\begin{aligned} H_0: &\quad \mu = 42 &\iff &\quad \mu - 42 = 0 \\ H_A: &\quad \mu > 42 &\iff &\quad \mu - 42 > 0 \end{aligned}$$

We then performed a single-tail single-sample t-test, and obtained $p = 0.043$ in Eq. (22.6). Remember what this p-value means:

> TRUE: p is the probability that our data would be as "extreme" as it is, assuming H_0 is true.

Reminder that by "extreme" we mean how far the data is from what's expected if H_0 is true. For example, in a paired t-test one would normally assume $H_0 : \mu_A - \mu_B = 0$, meaning that if H_0 were true, the expected value of t is zero. If we got $t = 1$, then the p value for a two-tailed test would be the probability of drawing data and getting $t > 1$ or $t < -1$ under the assumption of H_0.

At this point it is very commonplace for people to state (or imply) the following claims, all of which are FALSE:

1. FALSE: p is the probability that H_0 is true.

   This is false because frequentist probability only applies to data, and never to hypotheses or parameter values. It's tempting to try and "invert" things and make probability statements about the hypotheses, which is really what we want, but we need a Bayesian analysis to do it.

   The matter is put succinctly by the American Statistical Association when they say the p-value *"is a statement about data in relation to a specified hypothetical explanation, and is not a statement about the explanation itself."* in *The ASA Statement on p-Values; Context, Process, and Purpose.* [10]

2. FALSE: $1 - p$ is the probability that the opposite of H_0 is true.

   This is false for the same reason. We can't assign probabilities to hypotheses.

3. FALSE: A low p-value is strong evidence against H_0.

   This is false because we aren't comparing the probability against anything. It could easily be that our data is unlikely under *any* hypothesis, because unlikely things happen all the time.

   As stated in the ASA Statement on p-values, *"By itself, a p-value does not provide a good measure of evidence regarding a model or hypothesis."* [10]

4. FALSE: A low p-value is strong evidence in favor of the opposite of H_0.

   This is false for the same reason: the data might be unlikely for many different hypotheses, and it is illogical to single out H_0.

5. FALSE: A high p-value is strong evidence in favor of H_0.

   This is false because, again, we aren't comparing the probability against anything. Just because our data is very probable assuming H_0 doesn't logically preclude the data from being probable under other hypotheses. For example, consider the hypothesis H_0 : "Santa Claus exists". Your data is a missing plate of cookies near the fireplace on Christmas morning. If H_0 is true, then the data is actually quite likely. You'd get a high p-value. Does this high p-value provide strong evidence that Santa Claus exists? It obviously doesn't, because there are plenty of *other* hypotheses also highly compatible with the data. Another hypothesis might be "Dad loves cookies and eats them after the kids go to bed", and this hypothesis also results in a high p-value.

6. FALSE: A high p-value is strong evidence against the opposite of H_0.

   This is false for the same reason. If a high p-value doesn't provide strong evidence in favor of H_0, then logically it cannot provide strong evidence against the opposite of H_0.

All we're left with is that "p is the probability of getting data as extreme as we got, assuming H_0", and this doesn't tell us what we want to know. We want to know how likely the various hypotheses are, since these are the things we're uncertain of, but there's just no way to do it in a frequentist analysis that assigns probability to the thing we do know (the data). This is one reason why NHST is not an optimal tool for scientific research.

23.2 Issues With The General NHST Procedure

In the previous section we discussed how NHST doesn't tell us what we want to know. But there are other serious logical deficiencies as well:

In debates it is common to ignore a legitimate argument from an opponent, and instead invent a related but fake argument that is easy to rebut. This is known as creating a *straw man*. In effect, H_0 is a straw man, and rejecting it doesn't really tell us much.

1. A traditional null hypothesis like $H_0 : \mu = 42$ has no probability of being true to begin with. Note that μ is a *continuous* number, and hence there is zero probability that μ equals one specific, exact number down to an endless number of decimal places. Thus, H_0 *can be rejected with certainty on logical grounds.*

   To see this in physical terms, consider our fertilizer example. Clearly the fertilizer must have *some* effect on plant growth, because *everything* has an effect on plant growth. If we used sawdust instead of fertilizer it would have *some* effect, albeit probably tiny. Small variations in genetics, sunlight, temperature, soil composition, humidity, etc. will all affect plant growth, even if only a little bit. And the "average" value of these things will always differ a bit from group to group. Hence, in real research, two samples are *never* drawn from the exact same population.

2. Given that H_0 can't be true, what we really want to know is *how* different is μ from 42. If the fertilized tomato plants have a mean height of 42.05 inches, we wouldn't bother paying for the fertilizer. We want an *effect size*,

or some indication of *how much* the fertilizer improves growth. The NHST method doesn't answer this question, but instead gives a "yes/no" answer to another question we really don't care about and already know the answer to. If $p < \alpha$ it is typical for people to say that the impact of the fertilizer is `statistically significant`, but this term is misleading, because statistical significance is NOT the same thing as physical significance.

The ASA Statement on p-values states *"A p-value, or statistical significance, does not measure the size of an effect of the importance of a result."* [10]

3. The $\alpha = 0.05$ cutoff is totally arbitrary, and changing this arbitrary number means the same data may or may not show "statistical significance". Often people are tempted to change α *after* they compute p, just so that they can claim to have found statistical significance. This dubious behavior is discouraged, and often you'll hear people say "pick a significance cutoff α before you look at the data, and stick with it." That's good advice, but it's kind of beside the point: a sane method of scientific inquiry shouldn't result in reaching opposite conclusions on the basis of what you pick for some arbitrary number.

The ASA Statement on p-values states *"Scientific conclusions and business or policy decisions should not be based only on whether a p-value passes a specific threshold."* [10]

4. It's one thing to give a quantitative *measure* of compatibility of data with H_0, but it makes no sense to convert this to a "yes/no" statement, i.e. "significant" or "not significant". Statistical questions in science involve degrees of uncertainty and probabilities, not "yes/no" answers. It's okay to use probabilities to inform yes/no *decisions* by human beings, but to formulate scientific *claims* in yes/no form is utterly absurd.

Maybe when you get to claims like "The earth moves around the sun" or "Things are made of atoms" you can talk of scientific "facts" or talk of being certain. But if you're collecting data or performing a hypothesis test, it's almost assured that you don't have anything close to certainty, and it's bad to pretend otherwise.

5. It's kind of besides the point since H_0 is automatically false anyways, but it's worth repeating that failure to reject isn't logically the same thing as accepting. Thus, in a hypothesis test of this kind you can never actually accept the hypothesis you care about; instead, you have to pick some sort of "opposite" hypothesis and see if you can reject it. This is a big logical mess that confuses nearly everybody that tries to use this stuff in their research.

Note that a Bayesian analysis easily avoids all such issues. Model the data with a normal density with mean μ, assign a non-informative prior, compute the posterior density for μ using software like R and Stan, and then compute various probabilities such as $\mathbb{P}(\mu > 42)$ or $\mathbb{P}(\mu - 42 > 2)$, etc. You won't get a "yes/no" answer to the question of whether the fertilizer helps, but only probabilities, and that's correct.

Of course the Bayesian method isn't foolproof either, because you still need to make proper choices for the likelihoods and priors, but a Bayesian analysis doesn't have any of the numerous problems mentioned in this chapter.

In the NHST method, all you can say is that you've rejected $\mu = 42$ with "statistical significance" and report a p-value of 0.043. Furthermore, remember this p-value is nothing but a measure of the incompatibility of data with the null hypothesis you already know can't be true. Unfortunately, the NHST analysis doesn't answer the questions we want to ask, and instead offers partial answers to questions we don't care about. It's better to just compute an FCI, or even better perform a Bayesian analysis.

23.3 The p Value Lottery

While not a logical flaw of NHST itself, there is an even bigger problem with NHST: when combined with a social bias to only publish results that have "statistical significance", you end up with huge numbers of scientific papers reaching false conclusions.

This publication bias hasn't gone unnoticed, hence papers like *Why Most Published Research Findings Are False* by John Ioannidis [13].

To see how this works, consider studying the impact of a "fertilizer" that you already know is fake – it's just regular dirt. Assume that unfertilized plants are known to grow to an average height of 42 inches with standard deviation 6.5. Thus, the true mean height of plants getting the fertilizer is just $\mu = 42$, i.e. the fertilizer doesn't do anything. Imagine measuring 4 plants at random in the field, and computing a p-value under the null hypothesis $\mu = 42$, which is approximately valid here. What types of p-values would you expect? If the null hypothesis H_0 above is actually correct, then computing p-values from drawn samples is equivalent to drawing p-values from a uniform density (see Comment 23.1 below). In short, *p-values are uniformly distributed if H_0 is valid.* You could literally get any p-value with equal probability. It's a kind of "p-value lottery".

Comment 23.1 – Why is $p \sim$ uniform$(0, 1)$ (Optional Reading)

First note that a p-value of 1.00 means there was a 100% chance of getting data at least as extreme as what you got. Well, 100% means that *any* possible data is more extreme than this. Hence, $p = 1$ corresponds to zero deviation from μ, i.e. your sample mean was $\bar{h} = 42$ exactly. Equivalently, $t = (\bar{h} - \mu)/(s\sqrt{n}) = 0$ here. Note that there is zero chance of this, for the same reason as usual: p is a continuous number, and the probability to be a specific single number is always zero. I only get non-zero probabilities for *intervals* of p-values.

What would a p-value of 0.99 mean here? It means that there is a 99% chance that sample data would be at least as extreme as what I got. This sample must have a combination $t = (\bar{h} - \mu)/(s/\sqrt{n})$ that's a little bit aways from zero. How does the t statistic start to get farther and farther from 0? Well, some combination of $\bar{h}$ getting farther from μ, and the sample standard deviation s being smaller (so that you're dividing by a small rather than a big number). Note that any data more extreme than this $p = 0.99$ data will correspond to smaller and smaller p values. Thus, since there is a 99% chance of getting data at least as extreme as what we got here, there is also a 99% chance that one would obtain a p-value of 0.99 or smaller. But that means that there must be a 1% chance of getting a p-value of 0.99 or greater, i.e. $0.99 < p < 1.00$.

We can continue this line of reasoning: the probability of $0.99 < p < 1.00$ is 0.01, the probability of $0.98 < p < 0.99$ is 0.01, etc. Every equal-width interval for p has an equal probability of showing up when we collect a sample. Hence, p-values are uniformly distributed under H_0. From a more formal mathematical point of view, this conclusion follows directly from the definition of the p-value as essentially the inverse of the cumulative probability function of whatever statistic (e.g. t) forms the basis of the test.

23.3.1 Type I Errors - False Positives

The uniform distribution of p means that when H_0 is true (i.e. fertilizer has no effect), if 100 people do an experiment in which they draw a sample, then on average 5% of these experiments will find p-values less than 5%. Thus, even if the fertilizer has zero actual impact, 5% of the experiments would conclude that the fertilizer

has been shown to have a statistically significant impact on plant growth. Rejecting H_0 even though it is true is called a `Type I Error`. Having 5% of the hypothesis tests result in Type I errors wouldn't be a problem if all 100 people published their results, because then we would see that only 5% of the experiments find $p < 0.05$ and we would realize that probably the fertilizer does nothing. But what happens instead is that the 95% of people who don't find statistical significance are trained into believing they haven't discovered anything useful, and so they don't publish. After all, they didn't find "statistical significance", and who wants to bother publishing something "insignificant"? Thus, only the 5% who find $p < 0.05$ publish. If somebody then surveys the scientific literature, *every paper they read* says that the fertilizer works. Which is of course false.

This generalizes to a huge amount of scientific literature. Journals in economics, medicine, biology, environmental science, education, psychology, etc. are flooded with papers that claim some type of effect exists because they got a $p < 0.05$. And that's where the analysis often stops. What's typically missing?

- A sequence of different researchers attempting to *replicate* the experiment, and seeing what percentage of such experiments also find "statistical significance" in the effect.
- Any indication of the *uncertainty* of the p-value itself.
- An estimate (with uncertainties) of the actual effect size, i.e. *how much* does something affect or influence something else.
- Some theoretical basis for understanding or explaining the reported effect.

Thus, when H_0 is true, there is a good chance NHST will be misused (either by individuals or collectively by society) to claim the presence of an effect that doesn't actually exist.

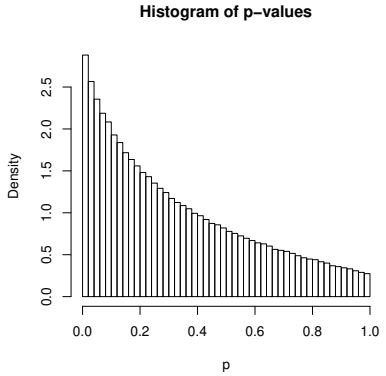


Figure 23.1: Histogram of 1e6 p-values computed under $H_0 : \mu = 42$ for size $n = 4$ samples drawn from a normal density with $\mu = 44, \sigma = 6$.

23.3.2 Type II Errors - False Negatives

What if H_0 is false? The distribution of p-values is no longer uniform, but it's still often quite wide, meaning that you could obtain a huge number of different p-values depending upon the sample you happened to draw. We still have a "p-value lottery". For example, imagine that our fertilized plants actually had a mean $\mu = 44$, which is a full 2 inches higher than 42. Further assume a $\sigma = 6$ for the fertilized plants. If we repeatedly drew samples of size $n = 4$, and performed NHST on $H_0 : \mu = 42$, what p-values would we obtain? Fig. 23.1 shows a histogram of the p-values obtained from one million samples simulated in R. It's clear that one could obtain an extremely wide range of p-values. For example, it's only 5 times more likely to get p near 5% as it is to get p near 50%. As such, there is a sizable probability of not rejecting H_0 when it's false, meaning that there actually *is* some type of effect present but the NHST misses it. Failure to reject H_0 when it is false is known as a `Type II Error`.

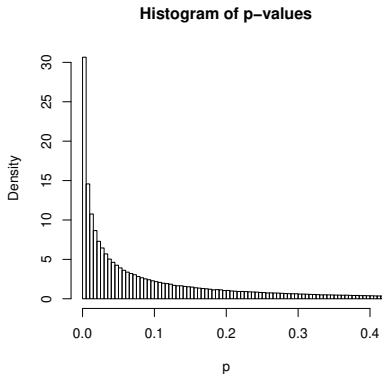


Figure 23.2: Histogram of 1e6 p-values computed under $H_0 : \mu = 42$ for size $n = 25$ samples drawn from a normal density with $\mu = 44, \sigma = 6$.

Simulations like this make it clear that the value of p is highly variable and doesn't really constitute strong evidence for or against most hypotheses. We can make things better by ramping up the sample size, but it's still a lottery. For the same situation with a sample size $n = 25$, the corresponding histogram of p-values is shown in Fig. 23.2. Only 49% of the p-values obtained in the simulation were less

than $\alpha = 0.05$, meaning that even for $n = 25$ there's basically a 50/50 chance you see a statistically significant p-value. Even though H_0 is false here, there is a 50% chance the NHST fails to reject it.

23.4 p Value Fishing

We end the chapter with a discussion of yet another problematic feature of NHST usage, known as p-value fishing. To understand this phenomenon, it is crucial to realize that in data sets with lots of different categories and types of measurements, it's almost always possible to find something that's *statistically significant*, even if there are no actual significant relationships. The reason is related to the discussion of the p-value lottery in Sec. 23.3. If you have measurements of 25 different variables, then there are $n = 25 \cdot 24/2 = 300$ different pairs of relationships between them. For $\alpha = 0.05 = 1/20$, on average 1 in every 20 of these relationships will have $p < \alpha$ assuming there is no actual relationship present in any of the pairs (i.e. if all null hypotheses are true). Hence, by random chance alone it is very likely that $p < \alpha$ for some pair, irrespective of whether the variables in the pair actually affect each other in real life.

According to the ASA Statement on p-Values, *"Proper inference requires full reporting and transparency. Conducting multiple analyses of the data and reporting only those with certain p-values renders the reported p-values essentially uninterpretable. Cherry-picking promising findings . . . leads to a spurious excess of statistically significant results in the published literature and should be vigorously avoided." [10]*

This practice of obtaining large datasets, testing for $p < \alpha$ in *every* relationship, and then publishing the "winners" is known as `p-value fishing` or `p-hacking`. It is a rather brazen Type-I error that can be made by a single researcher, without any need for the social complicity discussed in Sec. 23.3.1. The solution to this problem is quite simple: one should single out *specific relationships* to test *before* examining (or even collecting) the data.

To further illustrate the process of p-value fishing, we examine (and expand upon) an analogy due to physicist Richard Feynman.

23.4.1 Feynman's License Plate

By "every possibility is unlikely" we refer to specific outcomes. To flip a coin 10 times and obtain the exact sequence HHTHTTTTHT is pretty unlikely (a 1 in 1024 chance, in fact). However, "flipping 5 heads" isn't as unlikely, because there are actually lots of specific sequences (252, in fact) that give 5 heads, and you're now considering the chance of getting any one of these sequences. In essence, you're looking at the situation with a different "coarse-grained" perspective such that now there are only 11 possible outcomes (i.e. I can flip a total of 0, 1, 2, ..., 10 heads), in which case we're no longer talking about "a situation with a huge number of possible outcomes".

Let's start with the following observation: in a situation with a huge number of possible outcomes, it's pretty unlikely for any one particular outcome to occur (because it's competing with the million other things that could happen). But one of these unlikely things *will* happen, because *something* has to happen. There's no reason to be astonished or impressed with the fact that *something* happens, even if that particular thing is unlikely. It's not special. Join the club. *Every* possibility is unlikely. Richard Feynman once quipped

> You know, the most amazing thing happened to me tonight. I was coming here, on the way to the lecture, and I came in through the parking lot. And you won't believe what happened. I saw a car with the license plate ARW 357. Can you imagine? Of all the millions of license plates in the state, what was the chance that I would see that particular one tonight? Amazing!

The point is that it is unlikely to see any particular license plate. This fact does not turn around and make it somehow significant or special to see a specific "unlikely occurrance". Ultimately, probability is a method of quantifying plausibility and

uncertainty. But if we observe something, it's no longer uncertain. *It often makes no sense to ascribe special meaning to the probability of a specific outcome/event that has already happened*. What makes sense is to consider some special result *before* you conduct an experiment, and then see if it happens, because then you're testing *a specific* unlikely thing rather than simply looking for *any and all* unlikely things.

For example, after seeing "ARW 357", what if Feynman thought

> I always thought license plate numbers were totally random, i.e. each of the six characters is randomly drawn from the 26 letters and 10 digits (for a 1/36 chance of being selected). It's kind of funny that the plate I saw started with the letter "A". I wonder if maybe the plate numbers *aren't* random. Maybe plates are more likely to start with the letter "A". I should go observe a bunch of plates in the area and see if the fraction of plates starting with "A" is close to 1 in 36.

Proposing such an experiment/study makes sense. If Feynman then looked at 360 random licence plates, and 273 of them started with the letter "A", it would make sense to question the hypothesis "all plates are random". If, however, close to 1 out of 36 (i.e. 10) of the 360 plates started with "A", then there's no reason to think anything special is happening with "A". In either case, he could do further experiments to learn more.

On the other hand, what if Feynman observed his 360 random plates, and then thought the following:

> Well, if the starting letter/number was really random, then on average 10 of the 360 plates (i.e. 1 in 36) would start with "A". I found that 12 plates began with "A". This is more than 10, but there's always random variation, and so 12 isn't all that unlikely. On the other hand, a full 19 of the plates started with the letter "C". That's almost *double* what I would have expected on average. Thus, my experiment indicates that it's not the letter "A", but rather the letter "C" that is more likely, and hence random plate numbers aren't completely random.

It makes sense that Feynman wasn't impressed with the 12 plates starting with "A". But when he then pours over the data and notices something funny about "C", he's back to our original thinking error. It is almost certain that *some letter*, be it "C" or something else, would be overrepresented. The fact that indeed we found an overrepresented letter is not surprising or significant. We went into this experiment looking to test the letter "A". We didn't find anything unusual about "A", and now we should be done. Publish the result and go home. Have a drink. It's over. To pour over the data looking for "unlikely things" or "significant abberations" is just as meaningless as Feynman saying "seeing ARW 357 was really unlikely". You're virtually guaranteed to see *something* that's "unlikely". *If a data-set suggests some feature is unlikely, it makes no sense to use this very same data set to determine the significance of this feature.* You can't look at Feyman's data and conclude he's shown or inferred or deduced or proven *anything at all* about "C".

On the other hand, it would be perfectly okay for Feynman to say

> Well, it looks like nothing was funny about "A". But I sure counted a

lot more "C" than I expected. I should conduct another experiment to see if the counts of "C" are significantly different from 1 in 36. If I find an abberration in counts, I (and other people) should attempt to replicate the experiment, and see if we're on to something. But for now, all I've done is suggest that there doesn't (at this point) appear to be anything funny about "A".

Every time you pour over data and see something "funny" or "unlikely" or "significant", you're in danger of committing the above-mentioned thinking error. If your data suggests some weird or interesting thing *that you didn't even think of before the experiment/study*, and you wish to further investigate this interesting thing, then do additional experiments that are conceived and designed to investigate this specific interesting thing. It's of course okay if you aren't able to do these additonal studies; what's not okay is to naively or dishonestly suppose you can conduct a meaningful analysis of your interesting thing using the very same data-set that suggested the interesting thing. You'll just be fooling yourself and others. Be careful. This mistake *happens ALL the time* in NHST. It is hard to overstate how much ridiculous nonsense appears in published literature as a result of mining data for "unlikely relationships" (i.e. low p-values) and assuming that they're somehow significant in any meaningful sense.

Of course it's okay to point out the interesting thing you saw. Just be honest about it, and don't exaggerate. There's nothing wrong with publishing a statement like *"It's interesting that I got so many "C"s. Of course this could easily just be a fluke. After all, it's almost certain that* some *letter would be overrepresented. But perhaps it's something we could look at in future studies."* Just don't claim something meaningless like "statistical significance" to make your results look more impressive. Honesty is important.

23.5 Exercises

1. Find two peer-reviewed scientific papers that report results using p-values. Write a short summary (a paragraph or two) for each paper offering a critical evaluation of their use of NHST methods.
2. You compared two samples A and B of fish weights, and under $H_0 : \mu_A = \mu_B$ you found $p = 0.02$.
   i) Is $p = 0.02$ the probability that H_0 is true? Explain.
   ii) Does $p = 0.02$ constitute strong evidence that H_0 is false? Explain.
   iii) Is $1 - p = 0.98$ that probability that $H_A : \mu_A \neq \mu_B$ is true? Explain.
   iv) Does $1 - p = 0.98$ constitute strong evidence that H_A is true? Explain.
3. What is the difference between *statistical significance* and *physical significance*? Give an example to illustrate your point.
4. Imagine you have 6 different groups, and you wish to determine whether or not all of the groups are really drawn from the same population. To do this, you perform pair-wise t-tests on all the possible pairs of groups, using $\alpha = 0.05$.
   i) Assume that all null hypotheses are valid, i.e. $\mu_i = \mu_j$ for all groups i, j. When doing a single t-test on two of the groups, what is the probability of making a Type-I error? What is the probability $\mathbb{P}$ that you *don't* make a Type-I error?
   ii) How many different pairs N_p of groups are there?
   iii) If there is a probability $\mathbb{P}$ of not making a Type-I error on a single t-test, and you perform N_p of these tests in sequence, what is the overall probability of *not making* a Type-I error in the overall analysis? (Think about the basic rules of probability here).

iv) Using your previous answer, what therefore is the probability of *making* a Type-I error in the analysis?

v) Using the lessons of this problem, go back and evaluate the methodology employed in Ch. 22 Exercises 2 and 3.

If you do the problem properly, your answer to iv) should be over 50%. This is unacceptably high, and is the reason why repeated t-tests are a poor method with multiple groups. A preferred NHST method that controls the Type-I error rate is ANOVA, which we discuss in Ch. 25.

5. A researcher conducts a massive study of water quality in rivers. They select 100 different rivers in the US, and for each river they make measurements of local human population, ambient temperature, prevalent foliage, Pb concentration, e-coli levels, Hg levels, elevation above sea-level, average wind-speed, and so on. They pour over their data, and notice that there is a statistically significant ($p = 0.02$) impact of ambient temperature on e-coli levels for rivers that are surrounded by Maple trees, that have high Hg levels, and that are at low elevation. They share this with you, and ask your opinion about whether to publish the result. What would you say?

CHAPTER 24

NHST III: Power Analysis

In this chapter we discuss power analysis, which is a way of quantifying the impact of Type-I and Type-II errors (as discussed in Sec. 23.3) on our analysis. As we will see, for a t-test used to compare group means, the statistical power is the probability of detecting the presence of an actually-present difference between two group means. Power analysis will also offer us some guidance in choosing sample sizes when designing studies.

24.1 Classification of Inference Errors

Consider the problem posed in Example. 22.3. We specified our null hypothesis as

$$H_0: \quad \mu_A - \mu_B = 0 \quad \Longleftrightarrow \quad d\mu = 0 \tag{24.1}$$

though here we will consider a one-tailed test with the alternative hypothesis

$$H_A: \quad d\mu > 0 \tag{24.2}$$

Remember from Sec. 23.3 that there are two primary types of inference errors that can occur in NHST:

- A `Type-I Error` means that we rejected H_0 even though it was true. In other words, we're claiming to see an effect that isn't actually there. In short, it's a *false positive*. The discussion in Sec. 23.3 makes it clear that this happens a fraction α of the time, where α is the statistical significance level, often set to 0.05. Thus, we can also think of α as the Type-I error rate.
- A `Type-II Error` means we failed to reject H_0 even though it was false and H_A was true. We're failing to see some effect. It's a miss, or a *false-negative*. Conventionally the Type-II error rate is denoted β.

Whereas the value of α is something we choose, the value of β is an unknown, because the probability of missing some effect depends on the unknown effect size, which we now define.

24.2 The Effect Size and Cohen's d

Before computing β we first need to specify a possible strength of the effect we're looking for. One way of specifying the size of the effect would be the quantity $d\mu = \mu_A - \mu_B$. In our fertilizer example, this would be the increase in height one

gets from using the fertilizer. However, it ends up being more convenient to express the size of the effect in units of the standard deviation of one of the groups:

$$\text{effect size} = \frac{\mu_A - \mu_B}{\sigma_A} \quad \text{or} \quad \text{effect size} = \frac{\mu_A - \mu_B}{\sigma_B} \tag{24.3}$$

One reason it is convenient to express the effect size like this is that the result is dimensionless, meaning you'll have the same effect size no matter what system of units (say inches vs. cm) are employed.

Eq. (24.3) has two shortcomings. One, neither σ_A or σ_B are known ahead of time. Two, it's not obvious whether to use the σ from A or B. The solution to the first shortcoming is to use the sample standard deviations s_A and s_B. The solution to the second shortcoming is to use a value of s somewhere between s_A and s_B. For our fertilizer example, we have $s_A = 6.42$ and $s_B = 7.45$, so perhaps we could use the mean of these, i.e. $\bar{s} = 6.9$. It is more common, however, to compute an s from a properly weighted average of the individual sample variances called the pooled variance. The formula for the `pooled standard deviation` is

Using a weighted average means that the group with higher sample size counts more towards the value of s_{pooled}. Because $n_A = 4$ and $n_B = 9$, we expect the weighted average will be closer to s_B than s_A.

$$s_{pooled} = \sqrt{\frac{(n_A - 1)\, s_A^2 + (n_B - 1)\, s_B^2}{(n_A - 1) + (n_B - 1)}} \tag{24.4}$$

We can compute this for our current problem via

```
hA <- c(54.3,41.1,49.9,55.1)
hB <- c(43.6,46.8,33.9,51.5,33.4,31.5,48.2,40.3,48.1)
sA <- sd(hA)
sB <- sd(hB)
nA <- length(hA)
nB <- length(hB)
numerator <- (nA-1)*sA^2+(nB-1)*sB^2
denominator <- nA+nB-2
s.pooled <- sqrt(numerator/denominator)
```

(R.24.1)

In place of defining `sA` and computing `sA^2` we could instead just type `var(hA)` etc.

to obtain $s_{pooled} = 7.18$.

The difference $\mu_A - \mu_B$ expressed in multiples of s_{pooled} is called `Cohen's d`:

$$\text{effect size} = d = \frac{\mu_A - \mu_B}{s_{pooled}} \tag{24.5}$$

Equivalently, the difference in the two means is

$$\mu_A - \mu_B = d \times s_{pooled} \tag{24.6}$$

For example, an effect size of 0.2 would correspond to $\mu_A - \mu_B = 0.2 \times 7.18 = 1.44$ inches.

As stated earlier, the Type-II error rate β is the probability of missing an effect, i.e. of failing to reject $H_0 : \mu_A = \mu_B$ even when H_0 is false. This false-negative or "missing the presence of the effect" becomes less probable as the effect size d gets bigger. Conversely, if d is very small, there is a pretty good chance of missing the effect when analyzing the data, and β would be large.

24.3 Definition of Statistical Power

`Statistical power` is the probability that you reject H_0 given that the alternative hypothesis H_A is true. In other words, if H_A is true, then H_0 should be

rejected, and the probability that your test method actually does reject H_0 is known as the statistical power. If H_0 is false, then the effect size d is non-zero, and **the power of a hypothesis test is the probability it detects the presence of an effect** by rejecting H_0. Symbolically,

$$\text{power} = \mathbb{P}\left(\text{reject } H_0 | H_A \text{ is true}\right) \tag{24.7}$$

We can express power in terms of the Type-II error rate β as follows. There are only two possible outcomes for a hypothesis test: we reject H_0, or we fail to reject H_0. The probability of each possibility should add up to 1. Assuming H_A is true, we then have

$$\underbrace{\mathbb{P}\left(\text{reject } H_0 | H_A \text{ is true}\right)}_{\equiv \text{ power}} + \underbrace{\mathbb{P}\left(\text{fail to reject } H_0 | H_A \text{ is true}\right)}_{\equiv \beta = \text{ Type-II error rate}} = 1 \tag{24.8}$$

The first term is the definition of power, and the second term is the definition of Type-II error rate β, so we conclude that the power is just one minus the Type-II error rate:

$$\text{power} = 1 - \beta \tag{24.9}$$

This makes sense: assuming H_A is true, if β is the probability you miss the effect, then the power $1 - \beta$ is the probability you actually see the effect.

However, to repeat comments in the previous section, there is no way to compute β ahead of time, because the probability of missing an effect depends upon how big the effect is. The power $1 - \beta$ is something that can only be computed for hypothetical values of the effect size d.

The goal of conducting a power analysis on a hypothesis test is to relate sample sizes n, effect sizes d, significance level α, and statistical power $1 - \beta$.

24.4 Compute Power With R

The calculation of power is somewhat cumbersome (see Comment 24.1 below), but luckily it is performed automatically by the function `pwr.t2n.test()`, which requires installing and loading the `pwr` package. Let's assume that $\mu_A - \mu_B = 10$ inches. Recalling Eq. (24.5) and that $s_{pooled} = 7.18$, this corresponds to an effect size $d = 1.39$. For $n_A = 4$, $n_B = 9$, $d = 1.39$, and $\alpha = 0.05$, we can compute the power via

(R.24.2)
```
library(pwr)
pwr.t2n.test(n1=nA,n2=nB,d=1.39,alternative="greater")
```

which returns a power of 0.7. This is actually an approximate calculation, because these types of functions (in R and elsewhere) usually simplify the calculation by assuming $\sigma_A \approx \sigma_B$.

When dealing with equal sample sizes, instead of setting `n1` and `n2` to the same number in `pwr.t2n.test()` we can just use `pwr.t.test()`, which requires specifying only a single sample size parameter `n`.

The way `pwr.t2n.test()` works is as follows: there are 5 parameters you can specify: `n1`, `n2`, `d`, `sig.level` (which is just α), and `power`. You specify 4 of them, and the function will compute the one you didn't specify.

The output of `pwr.t2n.test()` is a list containing all of these parameter values, and you can use dollar-sign notation `$` to select one of them, as in Example 24.2 below.

We have focused on testing group means μ_A and μ_B, but one can do a power analysis in any situation involving NHST, including: regression models, chi-squared goodness of fit tests, comparisons of group variances, etc. See [14] for info on how to use the `pwr` package in some of these situations.

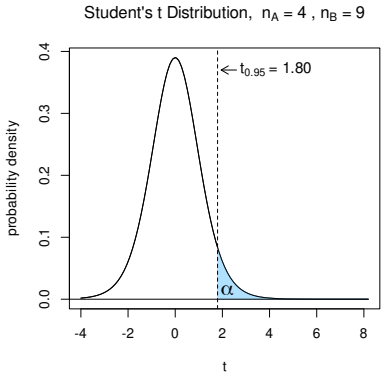


Figure 24.1: Central t distribution for $(\bar{x}_A - \bar{x}_B)/s_{pooled}\sqrt{1/n_A + 1/n_B}$ for $n_A = 4$, $n_B = 9$.

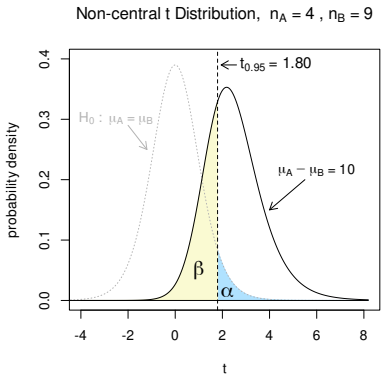


Figure 24.2: Non-central t distribution under $\mu_A - \mu_B = 10$ for $n_A = 4$, $n_B = 9$.

Comment 24.1 – Computing Power (Optional Reading)

To understand what `pwr.t2n.test()` is doing in (R.24.2), we first note that under the assumption $\sigma_A = \sigma_B$ we don't need to do the full Welch's t-test of Eq. (22.11). Instead, one can show [15] things simplify to

$$\frac{\bar{x}_A - \bar{x}_B}{s_{pooled} \cdot \sqrt{1/n_A + 1/n_B}} \sim t_{n_A+n_B-2} \tag{24.10}$$

where the degrees of freedom is $n_A + n_B - 2 = 11$. The 0.95 quantile of this t distribution $t_{0.95} = 1.80$ gives the cut-off value for t, i.e. values of t larger than this would result in p-values less than $\alpha = 0.05$ and hence we would reject H_0. This value of t is indicated in Fig. 24.1.

However, we recall that H_0 is not true here, i.e. we have assumed $\mu_A - \mu_B = 10$ (or equivalently $d = 1.39$). In this case, the distribution of the quantity on the LHS of Eq. (24.10) is no longer described by a t density centered at $t = 0$, but instead by a different type of distribution known as a *non-central* t-distribution, which is centered about a number known as the *non-centrality parameter*, or ncp for short:

$$ncp = \frac{d}{\sqrt{1/n_A + 1/n_B}} = \frac{\bar{x}_A - \bar{x}_B}{s_{pooled}\sqrt{1/n_A + 1/n_B}} \approx 2.3 \tag{24.11}$$

This distribution is shown in Fig. 24.2. Don't get too wrapped up in the exact details of this distribution; the important idea here is that there is a true distribution of $(\bar{x}_A - \bar{x}_B)/\sqrt{s_p \cdot (1/n_A + 1/n_B)}$ that differs from the null hypothesis distribution.

Now we come to the central point. In order for us to commit a Type II error, our sample must result in a value of t that is to the left of the dotted line $t = 1.80$ in Fig. 24.1, i.e. to the left of the blue-colored tail w/ area α. Remembering that the true distribution is the one in Fig. 24.2, the probability β of a Type II error is therefore the area to the left of $t = 1.80$ under the non-central t distribution, which is shaded in yellow. Continuing from (R.24.1),

```
t.95 <- qt(0.95,df=nA+nB-2)
d <- 10/s.pooled
ncp <- d/sqrt(1/nA+1/nB)
beta <- pt(t.95,df=nA+nB-2,ncp=ncp)
```

which gives $\beta \approx 0.3$ and hence power $= 1 - \beta = 0.7$, agreeing with (R.24.2).

Example 24.1: Minimum Detectable Difference

Problem: Continuing with the fertilizer example, what is the smallest value of $\mu_A - \mu_B$ that is detectable with 80% power at $\alpha = 0.05$?

Solution: We calculate the effect size d corresponding to power=0.80 via

```
pwr.t2n.test(n1=nA,n2=nB,power=0.8,alternative="greater")
```

which returns $d = 1.85$. We now need to convert this into a difference of

means $\mu_A - \mu_B$. If we are *designing* a research study, and hence have no data, we will need to guess a value for the population standard deviation σ. For example, if the σ for our tomato plants has historically been between 5 and 10, we could guess $\sigma \approx 7.5$ and then use Eq. (24.3) to compute

$$\mu_A - \mu_B = d \cdot \sigma = 1.85 \cdot 7.5 = 13.9 \text{ inches}$$

On the other hand, if we have already gathered data, we can use Eq. (24.6):

$$\mu_A - \mu_B = d \cdot s_{pooled} = 1.85 \cdot 7.18 = 13.3 \text{ inches}$$

This gap between μ_A and μ_B, for $n_A = 4$, $n_B = 9$, $\alpha = 0.05$, and power=0.8, is known as the `minimum detectable difference` between the two group means. If the difference in population means is less than around 13 inches, our t test will have less than an 80% probability of identifying the presence of the effect via rejection of H_0.

We again note that our result here is approximate in another way: `pwr.t2n.test()` assumes $\sigma_A \approx \sigma_B$, which we don't know is true.

Example 24.2: Plot Power vs. Minimum Detectable Difference

Law and Order Criminal Intent - S02.E0 **Problem:** Continuing with Example 24.1, make a plot of power vs. minimum detectable difference at $\alpha = 0.05$.

Solution: Assume `s.pooled` has been computed as 7.18. Let's make a plot for $\mu_A - \mu_B$ ranging from 0 to 20 inches using `curve()`, where we will let `x` denote $\mu_A - \mu_B$. Note that `pwr.t2n.test()` expects an effect size, so we will let `d=x/s.pooled`. Also, we want the power as a percent, so we'll multiply by 100.

```
curve(expr=100*pwr.t2n.test(n1=nA,n2=nB,d=x/s.pooled,
                            sig.level=0.05)$power,
      xlim=c(0,20),
      main=expression(paste("Power vs. ", mu[A]-mu[B])),
      xlab=expression(paste(mu[A]-mu[B], " (inches)")),
      ylab="power (%)",
      panel.first=grid())
```

Reminder that `plot()` usually expects vectors of x and y values. Since we are here trying to plot a smooth curve, i.e. the output of `pwr.t2n.test()`, it is a bit easier to use `curve()`.

The graph is shown in Fig. 24.3.

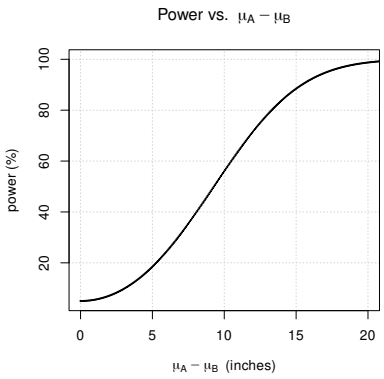


Figure 24.3: Power vs. minimum detectable difference at $\alpha = 0.05$ for $n_A = 4$, $n_B = 9$.

This plot provides a nice visual summary of the ways in which Type-I and Type-II errors place limitations on what we can conclude from our t-test. For example, we see that if the fertilizer causes a growth increase of around 20 inches, i.e. if $\mu_A - \mu_B \approx 20$, then there is nearly a 100% probability that the t-test detects the effect via rejecting H_0.

On the other hand, if the fertilizer enhances growth by around 5 inches, there is only around a 20% probability (power=0.2) that the t-test detects the effect. In other words, at a minimum detectable difference of 5 inches, the Type-II error rate β is around 80%.

If we require our t-test to have more statistical power at $\alpha = 0.05$, the only thing we can really do is collect more data. We discuss the impact of sample size on statistical power in the next section.

24.5 Determining Required Sample Sizes

A key question in designing a study or experiment is *"how big should the sample sizes be?"* Statistical inference tends to improve as sample sizes increase. However, external factors (available samples, time, money, etc.) place practical limitations on attainable sample sizes. How do we know if a sample size is *sufficient*? And what does *sufficient* mean here, exactly?

A common approach to answering these questions (in the context of frequentist methods) is to determine the sample sizes needed for a minimum detectable difference at some specified power and significance level.

Example 24.3: Computing Required Sample Size

Problem: You are planning a study in which you will compare two groups with unknown population means. Prior experience shows these populations tend to have a standard deviation of around 7. You plan on running an $\alpha = 0.05$ t-test on the samples. What sample size is needed for the t-test to have a 90% probability of detecting $|\mu_A - \mu_B| > 6$?

The use of absolute value bars here means that either $\mu_A - \mu_B > 6$ or $\mu_A - \mu_B < -6$, i.e. we are specifying a two-tailed test. If we wished only to consider a positive minimum detectable difference $\mu_A - \mu_B > 6$, we could set the `alternative` parameter to `"greater"` in `pwr.t.test()`.

Solution: Since we haven't taken sample data yet, we can't compute the pooled sample standard deviation. However, we can approximate the effect size via Eq. (24.3) as

$$d \approx \frac{\mu_A - \mu_B}{\sigma} = \frac{6}{7} \approx 0.86 \tag{24.12}$$

It is typically preferable to have the sample sizes match, and as such we can use `pwr.t.test()`:

```
pwr.t.test(d=0.86,sig.level=0.05,power=0.9)$n
```

Reminder that the output of `pwr.t.test()` is a list with the elements `n`, `d`, `sig.level`, and `power` among the contents. You can use the dollar-sign notation to select elements from a list, e.g. `$n` to only return the sample size.

which returns $n = 29.4$. Thus, each sample should contain 30 measurements in order for there to be a 90% probability of rejecting H_0 and hence detecting a statistically significant effect.

In the example above we chose 90% as an acceptable power. This of course means there is a $\beta = 10\%$ chance of committing a Type-II error, i.e. failing to detect the existing difference in μ_A and μ_B via rejecting $H_0: \ \mu_A = \mu_B$.

Of course it would be lovely if we could somehow arrange for there to be a 0% Type-II error rate, but this would require infinite sample sizes. Sometimes you can measure *all* of the available samples and there's nothing to decide. Usually you must choose n, and you'll just have to make a judgment call regarding the statistical power desired for possible minimum detectable differences.

Example 24.4: Plot Sample Size vs. MDD

Problem: Continuing with Ex. 24.3 above, make a plot containing curves of required sample size vs. the minimum detectable difference $\mu_A - \mu_B$ for a variety of different power levels.

Solution: Due to the way that `pwr.t.test()` solves for the required sample size, we can't use `curve()` as we did in Ex. 24.2. We will instead use the always-powerful `lapply()` function, and then use a `for` loop to plot the curve for various values of the power. First, we'll pick the lower and upper values of the minimum detectable difference $\mu_A - \mu_B$ we wish to plot, and establish some tick marks for custom axes:

```
mdd.min <- 2
mdd.max <- 10
x.ticks <- seq(2,10)
y.ticks <- seq(0,100,10)
```

Next, we create an empty plot window:

```
plot(x=NA,y=NA,
     xlim=c(mdd.min,mdd.max),
     ylim=c(0,100),
     xlab=expression(mu[A]-mu[B]),
     ylab="Required Sample Size",
     xaxt="n",yaxt="n",
     panel.first=abline(v=x.ticks,h=y.ticks,
                        col="gray90",lty="dashed"))
```

Then we add custom axis tick marks:

```
axis(side=1,at=x.ticks)
axis(side=2,at=y.ticks)
```

Now we'll define the power levels we're interested in. We will also create a vector of values for $\mu_A - \mu_B$ that we will use to compute sample sizes for. We'll also create a vector of colors for the different curves.

```
power.vals <- c(0.6,0.7,0.8,0.9,0.95,0.99)
num.curves <- length(power.vals)
mdd.vals <- seq(mdd.min,mdd.max+1,0.05)
d.vals <- mdd.vals/7            # pop. standard dev = 7
col.vals <- rainbow(num.curves)
```

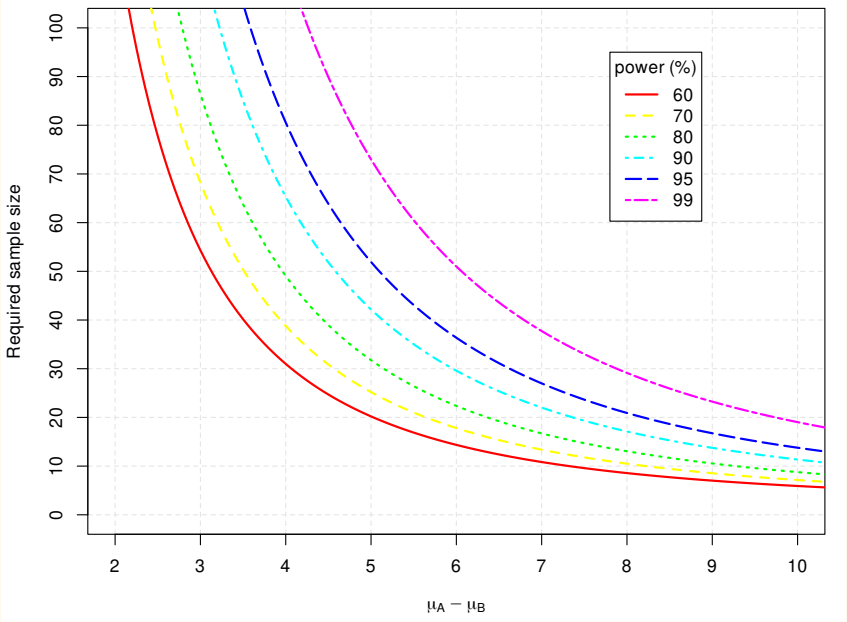
Next we use a `for` loop. For each value of the power, we will create a curve of sample size vs. $\mu_A - \mu_B$ using `lapply()`:

```
for (i in 1:num.curves) {
  n.vals <- lapply(d.vals, FUN = function(x)
                   pwr.t.test(d=x,power=power.vals[i])$n)
  lines(x=mdd.vals,y=n.vals,col=col.vals[i],lty=i,lwd=2)
}
```

Finally, we add a legend to identify the curves:

```
legend(x=7.8,y=95,title="power (%)",legend=power.vals*100,
       col=col.vals,lty=1:num.curves,lwd=2)
```

The final graph is shown below.



To use this graph to decide on sample sizes, first pick out a possible value of $\mu_A - \mu_B$, say 5. Focusing on the vertical line at 5, the red curve (power=0.6) indicates we'd need $n \approx 20$, the purple curve (power=0.99) indicates we'd need $n \approx 72$, and so on. In this way, we can investigate different possible effect sizes and see the sample sizes required at various powers.

This kind of graph can help you minimize the risk of collecting too little data in a research study. In fact, this type of analysis can provide some general guidance even if conducting a Bayesian analysis!

24.6 Exercises

1. A student conducts a class research project to compare the lengths of northern pike in two different rivers A and B. They measure the lengths of $n_A = 9$ pike in river A, and $n_B = 13$ in river B. They compute the mean and standard deviation in each sample, with the following results (all in cm):

$$l_A = 45.5 \qquad s_A = 3.3$$
$$l_B = 49.7 \qquad s_B = 7.1$$

   Assuming the lengths were drawn from normal populations, they conduct a two-tailed t-test of $H_0 : \mu_A = \mu_B$ at $\alpha = 0.05$, and find that $p = 0.077$. Thus, the student failed to reject $H_0 : \mu_A = \mu_B$. They then conclude that the two pike populations are identical (with regard to length).

   i) Does their conclusion follow logically from the result of the t-test? Explain. *Think carefully about the issues raised in Ch. 23 when answering.*

   ii) The failure to reject H_0 may have been a Type-II error. What is the probability β of a Type II error for this problem, assuming the true difference in means was 4 cm?

iii) What is β if the true difference in means was instead 6cm? 2cm?

iv) Use your values of β to offer an assessment of the adequacy of the student's research design.

2. Consider the corn data for fields A and B from Tab. 19.1. In the Exercises in Sec. 22.5 you performed a NHST for $H_0: \ \mu_A = \mu_B$. Consider the alternate hypothesis $H_A: \ \mu_A \neq \mu_B$.

   i) Make a plot of statistical power vs. effect size, assuming $n = 20$ samples.

   ii) What effect size gives a statistical power of 0.90? What value of $\mu_B - \mu_B$ does this equate to? (To answer this question you'll need to make an educated guess for σ. Justify your guess.)

   iii) Now assume that the sample size is variable. Make a plot of sample size vs. effect size for power = 0.7 and 0.9.

   iv) What sample sizes are needed for 70% power when $\mu_A - \mu_B = 2$? At 90% power?

3. You are designing a study to compare Pb (lead) levels in two separate lakes. Prior experience has shown that σ for Pb concentration in a lake tends to be around 4 ppb (parts per billion). You plan on using a t-test.

   i) For a minimum detectable difference in population mean of 3 ppb, with 80% power at $\alpha = 0.05$, what sample sizes are required?

   ii) Redo i) with power = 0.9.

   iii) Redo i) and ii) with a minimum detectable difference of 1 ppb.

   iv) Redo i), ii), and iii) with $\alpha = 0.01$.

   v) How much do your answers change if you had assumed a different value for σ, say $\sigma = 6$ ppb?

CHAPTER 25

NHST IV: ANOVA

25.1 Comparing Groups

In Chs. 17 and 19 we showed how to compare groups of various kinds with a Bayesian analysis. An alternative method of group comparisons, based on NHST, is called ANOVA. In some cases neither a Bayesian analysis nor ANOVA is necessary, because the groups are *obviously* different. For example, Susan is interested in finding out if there is a difference between two types of plant food: Miracle Sprout and Grow Huge. She decides to test the fertilizers on a specific bean plant. Four plants were treated with each, over a period of 6 days. The resulting plant heights are shown in Table 25.1.

i	fertilizer	h_{ij}				$\bar{h}_i$
1	control	2.42	4.99	5.69	5.69	4.70
2	Miracle Sprout	7.36	4.78	4.22	3.50	4.97
3	Grow Huge	11.3	10.3	17.2	17.6	14.1

Table 25.1: Bean plant height at 6 days.

Note that the right-most column indicates the mean of the data for each fertilizer, and isn't a datum value itself. The sample size for each fertilizer is 4. A similar display format is used in Tab. 25.2.

Note that each treatment group (i.e. the fertilizer type) represents a rather small sample of size $n = 4$. As such, we would expect a decent amount of variation *within* each group, which is observed. It is nonetheless clear, however, that the Grow Huge plants are significantly taller than the rest of the plants. The average height of 14.1 is nearly three times as the other two group means. It makes sense to conclude that group 3 is a sample drawn from a population different from that of either group 1 or group 2.

It's worth mentioning that we don't know for sure *why* the groups are different. Probably we're assuming the difference is just due to the fertilizers, but maybe better soil was used in the third group, or maybe somebody forgot to water the first two groups, or maybe the four healthiest seeds happened to be picked for the third group, etc.

But what if it isn't so obvious? What if the means between the groups are much closer? For example, the means are different between the control group and the Miracle Sprout group. So does Miracle Sprout actually have an effect? How big is it? How certain of it are you? When things become non-obvious, we can make use of ANOVA (or preferably a Bayesian analysis).

25.2 Principle Behind ANOVA

ANOVA, an pseudo-acronym for ANalaysis Of VAriance, is the comparison of variation *within* groups to variation *between* groups, in order to figure out if the groups are really different.

A simple example illustrates the general idea. Frank is curious and wonders if the

average height of male humans is different in the cities of Chicago, Bismarck, and Portland. Frank knows five people in each city, so he calls them up and asks them for their heights. He computes the mean sample height for each city. The data and means are presented in Table 25.2

i	City	h_{ij}					$\bar{h}_i$
1	Bismark	64.9	75.3	68.2	73.4	67.8	69.9
2	Chicago	66.0	73.8	63.0	69.8	67.2	68.0
3	Portland	74.5	68.7	76.9	76.0	69.3	73.0

Table 25.2: Heights in Various Cities (inches).

At first glance, it would seem that maybe people in Portland grow taller. For example, the Portland mean height $\bar{h}_P$ is about 3 inches taller than the Bismarck mean height $\bar{h}_B$. But then we look at the actual heights for Portland, and we see that the variation between the smallest and tallest height is over eight inches. Hence, the variation within Portland is bigger than the variation between Portland and Bismark. So maybe $\bar{h}_P$ is bigger than $\bar{h}_B$ due to random fluctuations? It's hard to be sure. So let's do the ANOVA analysis.

Sometimes in the ANOVA world groups are called *treatments*, which makes sense if you think of different groups of patients receiving different medical treatments. Of course in many situations the word *treatment* doesn't make sense as a synonym for *group*, but it's standard terminology.

ANOVA involves comparing the variation between groups to the variation within groups. Let $i = 1, 2, ..., k$ denote the different groups, and let h_{ij}, where $j = 1, 2, ..., n_i$ denote the heights in each group.

Step 1: Compute Group and Overal Means

The mean height in group i is computed as normal via

$$\bar{h}_i = \frac{\sum_{j=1}^{n_i} h_{ij}}{n_i} \tag{25.1}$$

with results shown already in Tab. 25.2. We can also compute an overall mean height for all the data. Let $N = \sum n_i = n_B + n_C + n_P = 18$ be the total number of height data. Then the overall mean is

$$\bar{h} = \frac{\sum_{i=1}^{k} \sum_{j=1}^{n_i} h_{ij}}{N} = 70.3 \tag{25.2}$$

Step 2: Compute Variation Between Groups

We now look at how much the group means $\bar{h}_i$ vary from the overall mean $\bar{h} = 70.3$. The `between groups sum of squares`, denoted here as SS_B, is defined as

$$\begin{aligned} SS_B &= \sum_{i=1}^{k} n_i \cdot \left(\bar{h}_i - \bar{h}\right)^2 \\ &= 6 \cdot (69.9 - 70.3)^2 + 6 \cdot (68.0 - 70.3)^2 + 6 \cdot (73.0 - 70.3)^2 \\ &= 79.5 \end{aligned} \tag{25.3}$$

We then compute the `between groups mean square`, denoted here as $\overline{SS}_B$, by dividing $SS.B$ by it's degrees of freedom. There are $k = 3$ groups, and remember we subtract one when computing variances, and thus

$$\overline{SS}_B = \frac{SS_B}{k-1} = \frac{79.5}{3-1} = 39.7 \tag{25.4}$$

Step 3: Compute Variation Within Groups

Next we compute the variation within groups. Because each group involves data presumably drawn from the same population, we assume any variation o a datum from the group mean is due simply to noise. Sometimes such noise in a datum is called its *error*. The `error sum of squares` or `residual sum of squares`, denoted here as SS_E, is computed by finding the sum of the residuals squared for each group:

$$\begin{aligned} SS_E &= \sum_{i=1}^{k} \sum_{j=1}^{n_i} \left(h_{ij} - \bar{h}_i\right)^2 \\ &= \sum_{j=1}^{n_B} \left(h_{Bj} - \bar{h}_B\right)^2 + \sum_{j=1}^{n_C} \left(h_{Cj} - \bar{h}_C\right)^2 + \sum_{j=1}^{n_P} \left(h_{Pj} - \bar{h}_P\right)^2 \\ &= 198.8 \end{aligned} \tag{25.5}$$

We next compute the `error mean square`, denoted here as $\overline{SS}_E$, by dividing by the degrees of freedom. Here there are N total data points (i.e. degrees of freedom), but k of them are effectively used up in computing the group means, leaving $N - k = 18 - 3 = 15$ degrees of freedom:

$$\overline{SS}_E = \frac{SS_E}{N-k} = \frac{198.8}{18-3} = 13.3 \tag{25.6}$$

Step 4: Perform The F Test

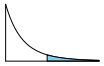
Under a null hypothesis that the data for each group are drawn from a common normal population, the ratio of $\overline{SS}_B$ to $\overline{SS}_E$ follows an F-distribution with degrees of freedom $k-1$ and $N-k$:

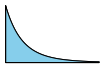
$$\frac{\overline{SS}_B}{\overline{SS}_E} = F_{2,15} = 2.998 \tag{25.7}$$

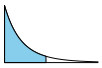
The probability of finding an F value at least this extreme is the area under the F-density to the right of 2.998. If F is sufficiently high, meaning the between-groups variation is a lot higher than the within-groups variation, then we'll conclude the groups aren't all from the same population. In other words, if F is high enough, we'll reject H_0 and conclude at least one of the groups is different than the others.

We can compute our p-value as

$$\begin{aligned} \mathbb{P}\left(F \geq 2.998\right) &= \text{area between 2.998 and } \infty \\ &= \\ &= \quad - \\ &= 1 - \texttt{pf(2.998,df1=2,df2=15)} \\ &= 1 - 0.92 \\ &= \boxed{0.08} \end{aligned} \tag{25.8}$$







Assuming the typical $\alpha = 0.05$, we have $p > \alpha$ and thus we fail to reject the null hypothesis. To conclude at least one of the groups is different (at $\alpha = 0.05$) we would have needed to see more variation *between* the different groups compared to the variation *within* the groups.

As a final comment, note that all our p-value does is reject or fail-to-reject H_0 at some level of statistical significance α. As usual, it doesn't tell us anything about the size of the differences. If we had high sample sizes, we could easily reject H_0 even if the actual effect size is very tiny. Remember that statistical significance isn't the same thing as practical or physical significance.

25.3 Doing It Simply With R

The `aov()` function in R will perform the ANOVA calculation automatically. To do so, we need store the data into a data frame. First, let's store the height data into one big vector `h`:

```
hB <- c(64.9,75.3,68.2,73.4,67.8,69.9)
hC <- c(66.0,73.8,63.0,69.8,67.2,68.0)
hP <- c(74.5,68.7,76.9,76.0,69.3,73.0)
h <- c(hB,hC,hP)
```
(R.25.1)

Next we'll create a big vector of group labels using the `rep()` function:

```
city <- rep(c("Bismarck","Chicago","Portland"),each=6)
```
(R.25.2)

We then combine `city` and `h` into a data frame:

```
height.data <- data.frame(city,h)
```
(R.25.3)

Following R's default behavior when creating a data frame, the categorical vector `city` will be converted (coerced) into a factor in (R.25.3). The command (R.25.4) says to group `h` data by the factor levels of `city` when performing the ANOVA computation.

Once we have our data frame, we can do the entire ANOVA analysis with the command

```
results <- aov(h ~ city, data=height.data)
```
(R.25.4)

whereupon `summary(results)` will return all the numbers we computed in the previous section.

25.4 Statistical Power

Like any hypothesis test, ANOVA has a probability β of making a Type-II error (i.e. failing to reject H_0 when it is false) and a corresponding probability $1-\beta$ (the power) of sniffing out the presence of a difference if it does exist. As before, β is not something that can be computed ahead of time, as it depends upon the effect sizes, i.e. the size of the differences in the group population means. We first need to generalize the effect size d to situations with more than two groups.

25.4.1 Effect Size in ANOVA

Assume that the population means μ_i are different for each city, but that they all share a common population standard deviation σ. We can define an effect size f

in ANOVA as follows. First we define an overall weighted mean

$$\bar{\mu} = \frac{1}{N}\sum_{i=1}^{k} n_i \mu_i \tag{25.9}$$

where (as before) N is the total number of data points and k is the number of groups (cities). A little algebra shows that this equation can also be written as

$$\sum_{i=1}^{k} n_i \left(\mu_i - \bar{\mu}\right) = 0 \tag{25.10}$$

meaning that the weighted sum of the residuals is zero. Next remember that we express effect sizes in units of the standard deviation s. When specifying s we either make a guess for the common population standard deviation σ, or we can compute a pooled variance that is the generalization of Eq. (24.4) for more than two groups:

$$s_{pooled} = \sqrt{\frac{\sum_{i=1}^{k}\left(n_i - 1\right) s_i^2}{\sum_{i=1}^{k} n_i - 1}} = \sqrt{\frac{\sum_{i=1}^{k}\left(n_i - 1\right) s_i^2}{N - k}} \tag{25.11}$$

where s_i is the sample standard deviation for each group. The effect size f in ANOVA is then defined as

$$f = \sqrt{\sum_{i=1}^{k} \frac{n_i}{N}\left(\frac{\mu_i - \bar{\mu}}{s}\right)^2} \tag{25.12}$$

where s is either s_{pooled} or a reasonable guess for σ, as was the case in Ch. 24. One can think of Eq. 25.12 as a generalization of Cohen's d in Eq. (24.5).

In Comment 24.1 we mentioned that when H_0 is false one deals with a non-central t-distribution with a specified non-centrality parameter. For the non-central F distribution arising here, the non-centrality parameter is $ncp = Nf^2$.

25.4.2 Choose Effect Size

Before we can compute power we need to specify a possible effect size. To be specific, let's (somewhat arbitrarily) choose

$$\mu_C - \bar{\mu} = -2 \tag{25.13}$$

$$\mu_P - \bar{\mu} = +3 \tag{25.14}$$

Once we pick these two, the third is specified by Eq. (25.10) as

$$\mu_B - \bar{\mu} = -1$$

Furthermore, we can use Eq. (25.11) on our actual data to compute $s_{pooled} = 3.64$. We finally use Eq. (25.12) to compute $f = 0.35$.

25.4.3 Use `pwr.anova.test()`

The function `pwr.anova.test()` from the `pwr` package has the following parameters:

- `k` = the number of groups.

- `n` = the sample size per group (assumed equal).
- `f` = effect size computed via Eq. (25.12).
- `sig.level` = significance level α.
- `power` = power $1 - \beta$.

As before, we specify 4 of these parameters, and it will solve for the one we didn't specify. Thus,

```
library(pwr)
pwr.anova.test(k=3,n=6,f=0.35,sig.level=0.05)
```
(R.25.5)

indicates that the power is $1 - \beta = 0.21$. A reminder of the interpretation: if the the true differences $\mu_i - \bar{\mu}$ were $-1, -2$, and 3, then there would only be a 21% probability of rejecting H_0 at $\alpha = 0.05$ and with sample sizes $n = 6$. Equivalently, the Type-II error probability here is 79%, which is rather high.

Playing around with different reasonable values for $\mu_i - \bar{\mu}$, we find that power remains pretty low. The problem is the small sample size. Assuming the same effect size $f = 0.35$, what sample sizes would we need for a power of, say, 80%? Assuming that s would remain near $s_{pooled} = 3.64$, or equivalently assuming that the common population standard deviation σ is around 3.64, we run

```
pwr.anova.test(k=3,f=0.35,sig.level=0.05,power=0.8)
```
(R.25.6)

to find $n = 27$. We would thus need at least 27 heights per city to reject H_0 with at least 80% probability for an effect size $f = 0.35$ at $\alpha = 0.05$.

Computing the effect size is somewhat cumbersome, but it is nonetheless important to conduct a power analysis when designing a study that will make use of ANOVA.

25.5 Comparison With Bayesian Analysis

The ANOVA analysis gave us a "no" to the "yes/no" question "do we reject H_0?" We can extract more information with a Bayesian analysis. Assume the height data for each city is drawn from a separate normal population, each with it's own mean and standard deviation. Let's also pick pretty wide weakly-informative priors:

$$
\begin{aligned}
h_i &\sim \text{normal}(\mu_i, \sigma_i) \\
\mu_i &\sim \text{uniform}(0, 100) \\
\sigma_i &\sim \text{uniform}(0, 30)
\end{aligned}
$$

Performing the normal Bayesian analysis with STAN, following the same steps as Ch. 19, we compute the posterior densities shown in Fig. 25.1. While it's true that there is some overlap in the densities, it still looks pretty clear that there are some differences. In fact, we can compute

$$
\begin{aligned}
\mathbb{P}\left(\mu_P > \mu_C\right) &= 0.95 \\
\mathbb{P}\left(\mu_P > \mu_B\right) &= 0.86 \\
\mathbb{P}\left(\mu_B > \mu_C\right) &= 0.75
\end{aligned}
$$

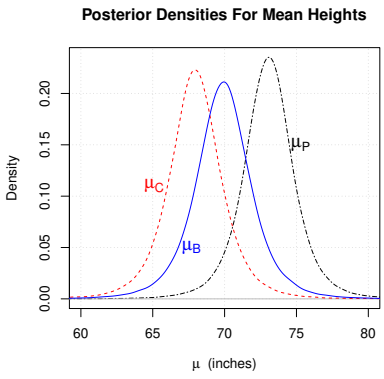


Figure 25.1: Marginal posterior probability densities for the mean population heights.

Are these probabilities sufficiently high that we conclude the group means are different? There's no unique answer, of course. From a pure scientific standpoint it's best just to give the posterior densities and various probabilities, and let people do what they want with the information. Boiling it down to a "yes/no" answer like we did in ANOVA doesn't really help, and the p-value we computed doesn't provide nearly as much information as a Bayesian analysis.

25.6 Other Types of ANOVA

This chapter outlined the simplest version of ANOVA, called *one-way* ANOVA. There are more advanced formulations of ANOVA that deal with more complicated subdivisions of groups, in order to tease out the impact of a variety of different factors that might be influencing the measurement of interest. We will briefly deal with such situations from another point of view in the last part of the book, where we examine regression analysis.

25.7 Exercises

Four different fertilizers were tested on a specific species of plant. For each of the four fertilizers, four plants were treated, and their mean height was later measured. The data is shown in the table below.

i	Fertilizer	x_{ij}				$\bar{x}_i$
1	No fertilizer	67	67	55	42	57.75
2	K_2O+N	98	96	91	66	87.75
3	K_2O+P_2O_5	60	69	50	35	53.50
4	N+P_2O_5	79	64	81	70	73.50
5	K_2O+P_2O_5+N	90	70	79	88	81.75

This data is taken from an example in [16], p529.

1. Create a properly formatted data-frame in R, and use the `aov` function to perform an ANOVA test of the null hypothesis that the treatment groups (i.e. different fertilizers) are all equivalent.

2. Compute the statistical power of the ANOVA test at your chosen α for an effect size of your choosing. What is the corresponding Type-II error rate?

   Note that to compute the effect size f you will first need to estimate reasonable values for $\mu_i - \bar{\mu}$ subject to Eq. (25.10).

3. Perform a Bayesian analysis of the same data. Include plots of the posterior densities, and also probabilities of the form indicated in Sec. 25.5.

4. Compare and contrast the strengths and weaknesses of the ANOVA and the Bayesian approach. In what ways are their conclusions similar, and in what ways do they differ?

Part V

Regression Analysis

CHAPTER 26

Linear Regression With One Predictor

A regression analysis with one predictor attempts to answer a very common scientific question: "how is A affected by B?" Typically A is affected by *many* other things (C, D, E, ...), so there tends to be a lot of variation (i.e. noise) in A not due to B, hence the need for probability and statistics (as discussed in Sec. 8.2). In this chapter we introduce regression analysis in the context of an extended example involving the impact of fertilizer on corn growth.

26.1 Framework of a Regression Analysis

26.1.1 Categorize Variation: Predictor(s) and Noise

Imagine that corn is grown in four fields, where a different amount of fertilizer x is applied to each field. If each corn plant had the exact same values of all factors that influence growth (soil composition, genetics, humidity, amount of fertilizer, temperature, water, sunlight, insect predation, etc.), then all corn plants would attain the same height. In reality, there is a lot height variation because all of these factors vary from plant to plant. Some sample data is shown in Tab. 26.1.

0	400	800	1200
101.8	135.8	135.6	129.2
105.2	114.6	111.2	122.6
107.4	134.1	119.2	130.3
109.6	107.2	104.6	148.2
126.5	109.9	128.5	144.1
99.8	111.4	97.3	99.6
125.0	137.3	121.5	136.3
106.0	112.6	131.5	127.7
111.4	117.1	115.5	143.3
119.2	107.9	120.0	149.1
112.2	133.0	137.6	128.8
120.8	124.7	118.9	133.1
125.7	126.3	111.2	147.2
101.8	120.6	136.4	125.6
104.9	131.4	128.0	133.1
129.9	133.9	122.7	149.8
117.0	129.3	112.6	141.9
120.5	92.4	123.5	121.5
130.3	107.0	136.2	126.9
107.4	107.9	135.6	115.4

Table 26.1: Corn height (inches) in four fields. The amount x of fertilizer (lbs/acre) for each field is listed at the top of the columns.

Let's separate all of this height variation into two categories: the variation due to the fertilizer, and the variation due to everything else (i.e. noise). When separated this way, the height of the corn is called a `response variable`, and the fertilizer is called a `predictor variable`.

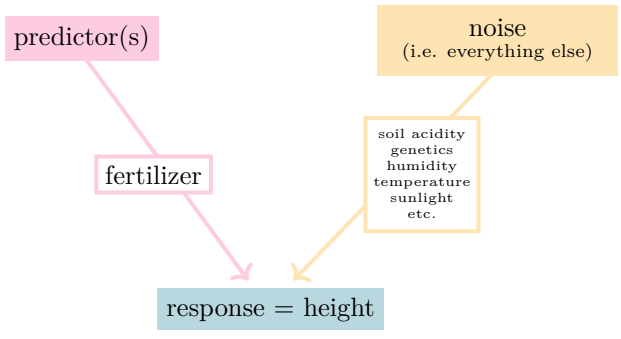


Figure 26.1: Physical causes of the height variation of corn plants split into two conceptual categories: predictors and noise.

A `regression analysis` seeks to establish the mathematical relationship between a response variable and some number of predictor variables. In our case, we seek the mathematical relationship between corn height and fertilizer amount.

In everyday usage the word "regression" means something like "to return to a previous condition". What does that have to do with the definition of *regression analysis*? Not much really. The word "regression" was first used in statistics by Francis Galton in the 1800s, based on a the specifics of a certain biological phenomenon [17]. "Regression" didn't make sense in a larger statistical context, but as often happens, the word stuck.

26.1.2 Pick a Distribution Type for Noise

In order to define a statistical model for our regression analysis, let's first assume that the height of the corn is described by the relationship

$$h \sim \text{normal}\big(\mu\,(x)\,,\sigma\big) \tag{26.1}$$

In other words, we are modeling a height measurement as a draw from a normal population. The mean $\mu(x)$ of this population depends upon the amount x of fertilizer used, and hence will be different for different fields.

The standard deviation σ is indicative of the variation in heights due to things other than fertilizer, i.e. noise. In other words, we are assuming the noise is normally distributed. We have further assumed that σ does not depend upon the amount of fertilizer, and is thus the same for all fields.

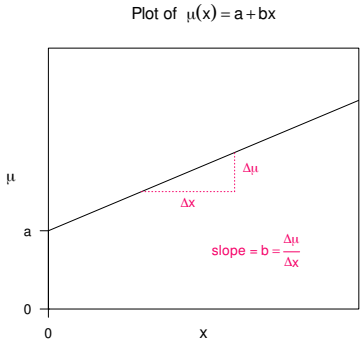


Figure 26.2: A hypothetical linear dependence of mean height μ on the amount of fertilizer x.

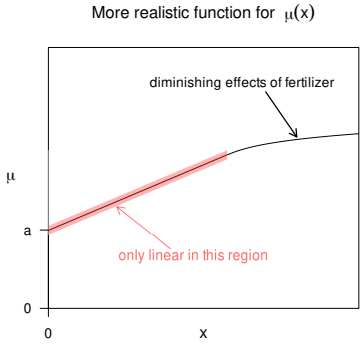


Figure 26.3: A more realistic relationship between μ and x.

26.1.3 Pick A Regression Equation

We now need to hypothesize a relationship between $\mu(x)$ and x. A good first guess is the straight-line relationship shown in Fig. 26.2, i.e. a *linear* relationship of the form

$$\mu(x) = a + b \cdot x \tag{26.2}$$

where a is the vertical intercept and b is the slope. The intercept a is the mean height in the absence of fertilizer (i.e. when $x = 0$). The slope indicates how much extra growth occurs for each additional unit of fertilizer. The bigger the slope b, the larger the impact of the fertilizer.

Taken literally, Eq. (26.2) says that you can make $\mu(x)$ as big as you want by simply adding enough fertilizer. Of course this isn't physically reasonable, and we conclude our equation could only be true for a limited range of fertilizer amounts. We expect that past a certain point, extra fertilizer won't improve plant growth (or might actually hinder it), as shown in Fig. 26.3.

It's worth emphasizing that at this point we have no reason to suspect that Eq. (26.2) is the correct regression equation. The graph of μ vs. x could be any kind of shape. But Eq. (26.2) is a reasonable initial guess. After conducting the regression analysis, we will evaluate our choice of regression equation. In later chapters we will consider the possibility of a non-linear regression equation.

26.1.4 Formulate Bayesian Statistical Model

Having specified a regression equation and a probability distribution for noise, we are ready to sketch out the full Bayesian statistical model. Note that there are three parameters in our analysis: a, b, and σ. Because these parameters are unknown, our uncertainty of them is encoded in a probability distribution. Our statistical model will take the general form

$$h_i \sim \text{normal}\big(\mu(x), \sigma\big) \tag{26.3}$$

$$\mu(x) = a + bx \tag{26.4}$$

We don't consider μ to be another independent parameter here, because once you know a and b and specify the amount of fertilizer x, then μ is given by Eq. (26.2).

where the parameters a, b, and σ are described by probability distributions that are to be determined. A schematic for the statistical model is indicated in Fig. 26.4 below:

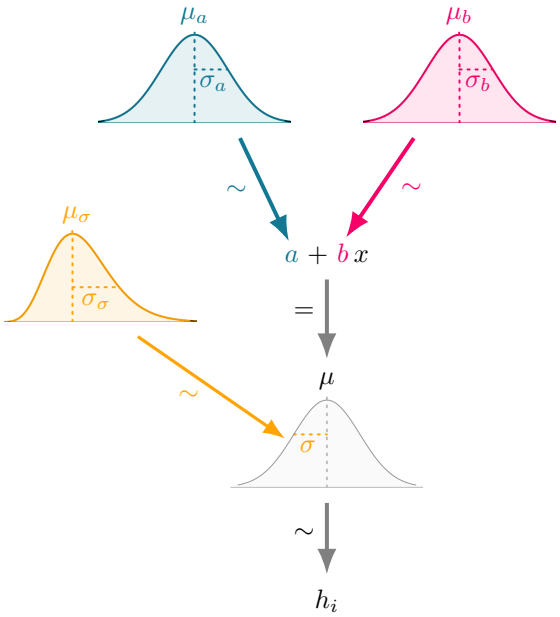


Figure 26.4: Schematic of Bayesian regression model.

In Fig. 26.4 the width of the probability distribution for h (the one feeding h_i at the bottom) arises from variation due to the noise factors indicated in Fig. 26.1. On the other hand, the widths of the probability distributions for the parameters a, b, and σ arise from the uncertainty in our knowledge of these parameters.

In this particular model note that a describes the mean corn height without fertilizer (i.e. at $x = 0$), and hence a is the *average* impact of all the factors comprising noise. The variation in h arising from *variation* in these noise factors is described by σ. The impact of the fertilizer is described solely by b.

It is also common for statisticians to define noise in a different way, as any variation in h not accounted for by the regression equation (in our case Eq. (26.2). Under this definition of noise, only σ (and not a) would be considered related to noise. In general, the term *noise* is somewhat imprecise insofar as its definition is context dependent.

26.2 Compute Regression Parameters

A Bayesian regression analysis seeks to compute the joint posterior for a, b, and σ via Bayes' theorem:

$$\rho(a, b, \sigma|D) = \frac{\rho(D|a, b, \sigma)}{\rho(D)} \cdot \rho(a, b, \sigma) \tag{26.5}$$

In the absence of strong prior information about these parameters, we should use weakly-informed / flat priors for the parameters. The function `stan_glm()` from the `rstanarm` package will choose such priors and generate draws from the posterior distribution.

26.2.1 Format the Data

Before using `stan_glm()`, we will need to create a data frame where each row gives the fertilizer amount and height for a certain corn plant. We first read in the data:

(R.26.1)
```
raw.data <- read.csv("corn_regress.csv")
```

We now demonstrate two different methods of getting the data into the appropriate tidy format (see 1.3.6).

Formatting Method # 1: Using `c()` and `rep()`

We first combine all of the columns into one large vector of height values:

```
h <- c(raw.data[,1],raw.data[,2],raw.data[,3],raw.data[,4])
```
(R.26.2)

Next we follow (R.25.2) in using `rep()` to make a vector of corresponding fertilizer amounts:

```
fert <- rep(c(0,400,800,1200), each=nrow(raw.data))
```
(R.26.3)

We can then combine these two vectors into our properly formatted data frame:

```
corn.dat <- data.frame(h=h, fert=fert)
```
(R.26.4)

Formatting Method #2: Using `stack()`

Reformatting data frames in various ways is also accomplished with the `melt()` and `cast()` functions of the `reshape2` library. We won't cover them here, but they are powerful functions that many people like using, so you may want to give them a look [18].

If we label the columns of the data frame, then the `stack()` function will treat these as factor levels and create a tidy data frame:

```
colnames(raw.data) <- c(0,400,800,1200)
corn.dat <- stack(raw.data)
colnames(corn.dat) <- c("h","fert")
```
(R.26.5)

Study the resulting data frame and make sure you know what `stack()` did.

While `stack()` is handy when the column headings are best regarded as categorical, in our case they are really numerical, i.e. fertilizer amounts. To properly perform our regression analysis, the fertilizer amount needs to be encoded as a number rather than a categorical factor level. We can follow the procedure of (R.5.31) to convert `fert` from a factor to a numerical vector:

```
corn.dat$fert <- levels(corn.dat$fert)[corn.dat$fert]
corn.dat$fert <- as.numeric(corn.dat$fert)
```
(R.26.6)

This is a little bit more work than the `c()` and `rep()` method for the current problem, though `stack()` is easier for larger data frames with many categorical columns. The remaining R code numbering assumes we used the first method.

26.2.2 Call `stan_glm()`

With the data formatted, we can now use `stan_glm()`:

```
library(rstanarm)
model <- stan_glm(h ~ fert, data = corn.dat,
                  iter = 2e4, cores = 4, chains = 4)
```
(R.26.7)

To inspect the priors chosen by `stan_glm()` run the command `prior_summary(model)`.

There are other parameters one can set to define informative priors, but we won't be using them here.

After running `stan_glm()`, you can get a quick summary of the results with the `summary()` function. Typing

(R.26.8)
```
summary(model,digits=3)
```

We need to set the `digits` parameter to at least 3, because the default value of 1 is not enough decimal places to see the value of the slope b.

returns the following parameter estimates:

(R.26.9)
```
Estimates:
              mean    sd      10%     50%     90%
(Intercept) 113.487   2.226 110.647 113.487 116.327
fert          0.015   0.003   0.011   0.015   0.018
sigma        11.872   0.962  10.691  11.808  13.134
```

Reminder that computations by Stan are approximate, and hence your numbers will be slightly different.

`stan_glm()` doesn't know anything about the symbol b we used to denote the slope. It uses `fert` to denote the coefficient of this predictor in the regression equation, i.e. the slope b.

The mean point-estimates for the parameters (denoted with a hat) are thus

$$\begin{aligned} \hat{a} &= 113.5 \quad \text{inches} \\ \hat{b} &= 0.015 \quad \frac{\text{inches}}{\text{lbs/acre}} \\ \hat{\sigma} &= 11.9 \quad \text{inches} \end{aligned} \tag{26.6}$$

Our uncertainty in these estimates is characterized by the `sd` column.

A nice graphical summary of the draws from the posterior density of the paramaters is obtained using the `pairs()` function:

(R.26.10)
```
library(hexbin)
pairs(model,
      pars=c("(Intercept)","fert","sigma"),
      diag_fun="hist",
      off_diag_fun="hex")
```

To see density curves instead of histograms on the diagonal, use `diag="dens"`.

with result shown in Fig. 26.5.

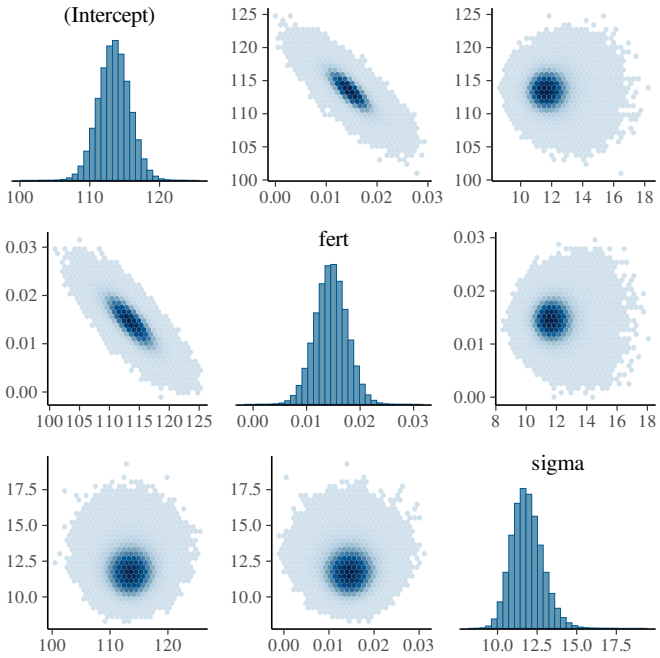


Figure 26.5: `pairs()` plot of draws from the posterior distribution of the regression coefficients a, b, and σ. This image was made with `iter=2e6`, `cores=16`, `chains=16`.

Note that when running a script, the command `pairs()` won't automatically show the graph. To get the graph to show, either run the command individually, or store the result using something like `result <- pairs(...` and then print it using `print(result)`.

The heat maps are essentially a top-view of the joint posterior plots (like the one in Fig. 18.3).

The diagonal images are histograms of the draws. The off-diagonal images require some explanation. Consider the image in the top-right. Each point on the plot indicates the values of the `(Intercept)` (aka a) and σ for a specific draw. The image is darker in regions where there is a higher density of points. Such an image is called a **heat map**.

The circular patterns in the plots involving σ are an indication that our knowledge of σ is not related to our knowledge of the other parameters. This is good for us, because we had assumed the the variation in heights due to noise was constant for all fields, i.e. independent of the amount of fertilizer.

On the other hand, the plots for a and b show that our knowledge for one depends upon our knowledge for the other. This is to be expected, because for a given height h if the impact of the fertilizer b is smaller, then the impact of a must be bigger to explain the data.

26.2.3 Extract Draws and Compute Credible Intervals

When using `stan_glm()` we don't use `extract()` to get the draws. Instead, we use `as.data.frame()`:

The quote marks are needed when extracting `(Intercept)` because otherwise the parentheses confuse R.

```
draws <- as.data.frame(model)
a <- draws$"(Intercept)"
b <- draws$fert
sigma <- draws$sigma
```
(R.26.11)

We can then compute Bayesian (even-tailed) credible intervals as usual:

```
CL <- 0.95
probs <- 0.5+c(-1,1)*CL/2
a.BCI <- quantile(a,probs=probs)
b.BCI <- quantile(b,probs=probs)
sigma.BCI <- quantile(sigma,probs=probs)
```
(R.26.12)

which gives the 95% intervals

$$a \in (109.1, 117.9) \quad \text{inches} \tag{26.7}$$

$$b \in (0.009, 0.020) \quad \frac{\text{inches}}{\text{lbs/acre}} \tag{26.8}$$

$$\sigma \in (10.2, 13.9) \quad \text{inches} \tag{26.9}$$

There is thus, for example, a 95% probability that b is between 0.009 and 0.020 inches/(lbs/acre).

We can also use the draws to make nicer versions of the parameter heat maps:

```
smoothScatter(a,b,nrpoints=0)
text(x=115,y=0.024,
     labels=expression(mu == a + b %*% fert))
library(car)
dataEllipse(a,b,plot.points=FALSE,add=TRUE,levels=0.95)
```
(R.26.13)

Note that you may need to first install the `car` (companion to applied regression) package via `install.packages()`. The resulting plot, with some extra formatting

and annotation added, is shown in Fig. 26.6. There is a 95% probability that the correct combination of a and b lies within the indicated ellipse. The point-estimate values for a and b are indicated by the dot in the center.

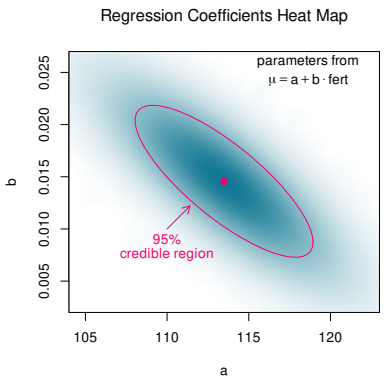


Figure 26.6: Heat map for the posterior draws of a and b, with a 95% credible region indicated. The point estimates for a and b are indicated by dot in the middle.

Finally, note that the parameter point estimates in (R.26.9) can be computed via

(R.26.14)
```
a.PE <- mean(a)
b.PE <- mean(b)
sigma.PE <- mean(sigma)
```

26.3 Evaluating the Model

26.3.1 Are The Sample Means Linear?

Let's create a plot containing the data points, the mean height for each field, and the point-estimate regression line $\hat{\mu}(x) = \hat{a} + \hat{b}x$:

(R.26.15)
```
plot(x=jitter(corn.dat$fert,factor=0.5),
     y=corn.dat$h,
     xlab="fertilizer (lbs/acre)",
     ylab="height (inches)",
     main="relationship between height and fertilizer",
     pch=19, col="gray", cex=0.5,
     panel.first=grid(lty="solid",lwd=0.5,col="gray90"))
abline(a=a.PE,b=b.PE,col="#127893")
x <- c(0,400,800,1200)
points(x=x,y=lapply(raw.data,mean),pch=19,cex=1.2)
text(x=310,y=143,pos=4,cex=1.3,col="#127893",
     labels=expression(hat(mu) == 113 + 0.015 * x))
arrows(x0=600,y0=140,x1=600,y1=122.7,
       length=0.1,col="#127893")
```

In Line 40 of (R.26.15) we use `lapply()` on the original data table as a quick way to find the means for each field. If the raw data is already in tidy format, i.e. in the form of `corn.dat()`, then we can use `aggregate()` as a quick way to compute these means: `aggregate(corn.dat,list(corn.dat$fert), mean)$h`. See Sec. 7.4.

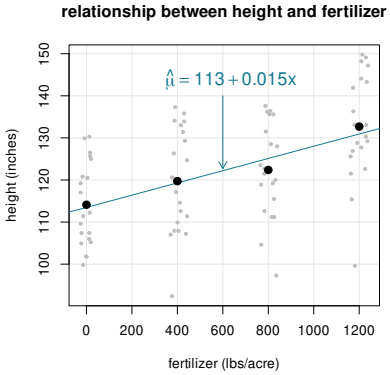


Figure 26.7: Plot of point-estimate regression equation. The larger black dots are the mean sample height for each field. A small amount of jitter was added to the data points to enhance visibility.

In Fig. 26.7 we see that the four sample means (one for each fertilizer amount) seem to lie pretty close to the regression line. The linear regression equation appears to describe the data pretty well.

We emphasize again that the regression equation $\mu = a + bx$ only describes the population mean μ for each field (i.e. fertilizer amount). There is also noise, meaning the individual data points will be scattered around the regression line.

26.3.2 Is The Noise Normal?

We have seen that $\hat{\mu}(x) = \hat{a} + \hat{b}x$ seems to do a good job of describing the sample means. An additional specification of our model is that the corn heights h_i are drawn from a normal distribution with standard deviation σ. Let's investigate whether this seems to be the case. We first discuss some terminology.

Errors vs Residuals

Our model for noise in Eq. (26.1) assumes that the difference between a datum

value h_i and the true mean $\mu(x_i) = a + bx_i$ can be considered as a draw from a normal distribution with mean 0 and standard deviation σ:

$$e_i \equiv h_i - \mu(x_i) = h_i - (a + bx_i) \sim \text{normal}(0, \sigma) \tag{26.10}$$

where e_i is often called the `error` of response value h_i. It is the error e_i which is the true measure of the noise.

However, because our point estimate $\hat{\mu}(x_i) = \hat{a} + \hat{b}x_i$ is an *estimate* and not the exact value of $\mu(x_i)$, it follows that we don't know the value of the error e_i. What we know instead is the value of the *residual* r_i, which is defined to be the difference between the datum value h_i and our point-estimate $\hat{\mu}(x_i)$:

This is analogous to our definition of the residual back in Eq. (2.2) as the difference between a datum value and the sample mean.

$$r_i \equiv h_i - \hat{\mu}(x_i) = h_i - \left(\hat{a} + \hat{b}x_i\right) \tag{26.11}$$

If our point-estimate line happened to equal the exact line, then the residuals match the errors, i.e. $r_i = e_i$. This is unlikely to happen, however, and in general our point-estimate line differs from the true line. At some locations i the point-estimate line moves closer to h_i than it should, and at other locations it moves farther away. Depending upon how much the noise in other locations drags the line toward or away from the true line, the residuals r_i will differ from the errors e_i. Thus, unlike the errors e_i, it is not strictly true that the residuals r_i are draws from normal $(0, \sigma)$. Nonetheless, for our purposes it will be adequate to assume the r_i provide estimates of the errors e_i, meaning we will approximate the r_i as being draws from normal $(0, \sigma)$.

A more accurate handling of the residuals involves a concept known as *statistical leverage*, which accounts for the degree to which different data points drag the best-fit line away from its true value. See [19] for some technical details.

The Q-Q Plot

First we express the residuals r_i in units of the estimated standard deviation $\hat{\sigma}$, i.e. as a z-score (see Sec. 14.4.3). These are known as `standardized residuals`.

$$z_i = \frac{r_i}{\hat{\sigma}} = \frac{h_i - \left(\hat{a} + \hat{b}x_i\right)}{\hat{\sigma}} \tag{26.12}$$

We can assess the normality of our standardized residuals by comparing them with the values $z_{i,\text{expected}}$ we're most likely to get from a standard normal distribution. If our z_i match up pretty closely with the expected values, then the points $(x, y) = (z_{i,\text{expected}}, z_i)$ should follow the 45-degrees line $y = x$. In other words, the actual and expected quantiles should match. This type of plot (comparing two sets of quantiles) is called a `Q-Q Plot`.

These "expected z_i" are computed by assuming that our n residuals r_i correspond to evenly spaced quantiles of a normal density. Consult the help file for `ppoints()` if you are interested in the details of the implementation.

A Q-Q plot can be easily generated in R with the function `qqnorm()`. We first need the residuals defined in Eq. (26.11). These are included in the `residuals` element of the variable `model` returned by `stan_glm()`. We convert these to standardized residuals and then call `qqnorm()`:

```
r <- model$residuals
z <- (r-mean(r))/sigma.PE
qqnorm(z,xlim=c(-3,3),ylim=c(-3,3),
       main="Q-Q Plot of Residuals")
abline(a=0,b=1)
```

(R.26.16)

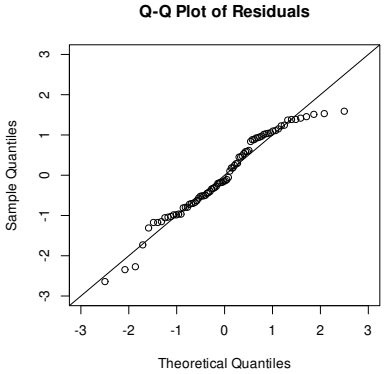


Figure 26.8: Q-Q Plot of the standardized residuals we obtained and those theoretically expected.

The points in the resulting Q-Q plot of Fig. 26.8 fall pretty close to the 45 degree line. Hence, the noise seems to be adequately described by a normal distribution. We aren't being specific about what "adequate" means, and simply note that if the points seem to be quite far removed from the 45 degree line, then perhaps the assumption of normally-distributed noise should be reconsidered.

26.4 Visualize Uncertainties in Regression Model

Of course we aren't certain that the point-estimates $\hat{a}$, $\hat{b}$, $\hat{\sigma}$ are correct – they are simply the most likely values.

Our uncertainty is captured by the posterior distributions for the parameters shown in Fig. 26.5. Because there is uncertainty in a and b, there is also uncertainty in the value of $\mu(x)$, i.e. there is a posterior distribution for $\mu(x)$ also. Fig. 26.9 shows the posterior distribution for μ at each of the four fertilizer values. Our point-estimate regression line hits the center of each of these distributions, but other regression lines are possible. As a challenge, see if you can reproduce Fig. 26.9.

It is very important to keep in mind that our uncertainty in the values of the regression parameters is not the same as the noise in the data. This is illustrated in Fig. 26.10 below.

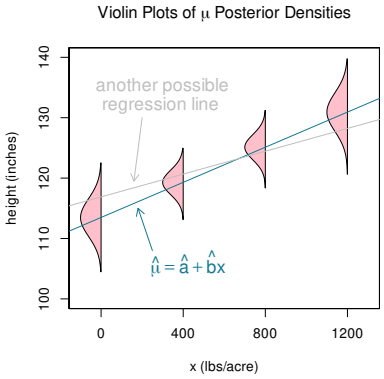


Figure 26.9: Violin plots of the posterior distribution for μ at the four fertilizer values x.

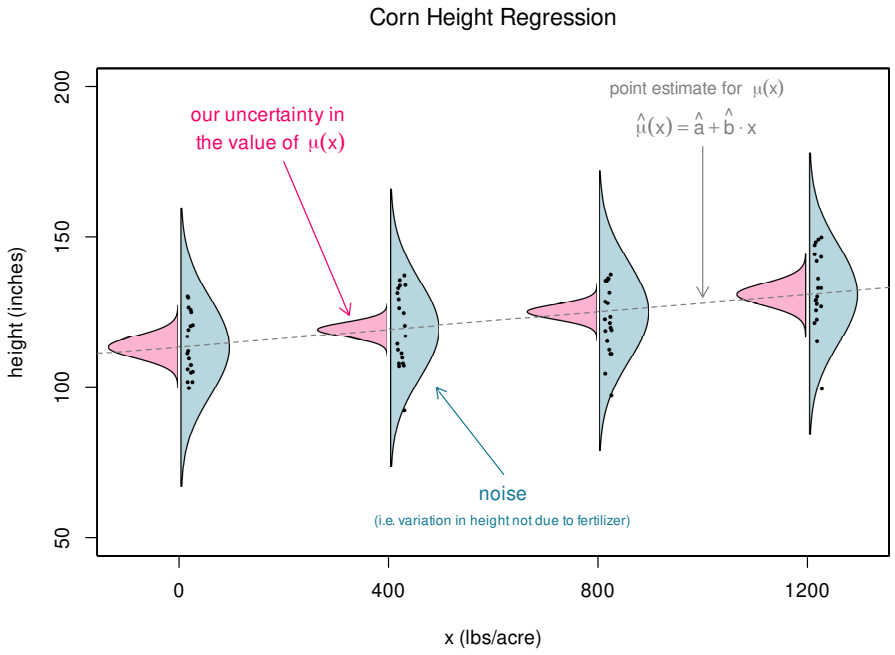


Figure 26.10: At each of our four fertilizer values, there are effectively two probability distributions at play in our regression model. First, there is a scatter of data points due to variations in the noise factors indicated in Fig. 26.1. This normal variation is shown to the right of each data point, in blue. Second, there is our uncertainty in the value of the regression paramaters a and b (and hence also in $\mu(x) = a + bx$), which is indicated by the pink distribution curve on the left.

In Figs. 26.9 and 26.10, the posterior for μ was shown only at 4 values of the fertilizer x. It will be useful to visualize our uncertainty in μ at *all* values of x.

At every value x we will use a function compute the 95% credible interval for μ and plot this interval as a shaded vertical line-segment at location x. This will create a shaded region around the regression line called an *uncertainty band*, here at the 95% level.

To begin, we define a function to compute and plot the μ credible interval at a fertilizer value x:

To attain a precise color and opacity for the uncertainty band, it is preferable to define big vectors of x and y coordinates of all the endpoints of the credible intervals, and then join all these points together into a single shaded polygon using the `polygon()` function. Doing so is a bit complicated, so we opt for a simpler (though cruder) method.

```
mu.BCI <- function(x) {
   mu.draws <- a+b*x
   BCI <- quantile(mu.draws,probs=c(0.025,0.975))
   lines(x=c(x,x),y=BCI,col=rgb(1,0,0,0.03),lwd=5)
}
```
(R.26.17)

In line 53 the number 0.03 in `rgb()` and the linewidth `lwd` can be adjusted to create the desired opacity of the band.

Next we create an empty (though labeled) plot:

```
plot(x=corn.dat$fert,y=corn.dat$h,
     xlab="fertilizer (lbs/acre)", ylab="height (inches)",
     main="relationship between height and fertilizer",
     pch=NA, xaxs="i", yaxs="i",
     panel.first=grid(lty="solid",lwd=0.5,col="gray90"))
```
(R.26.18)

Using `lapply()`, we will now call our custom function `mu.BCI()` on a large number of x-values. This will draw a thin vertical shaded interval at each x, and the result will be a shaded band:

```
lapply(seq(0,1200,3),mu.BCI)
```
(R.26.19)

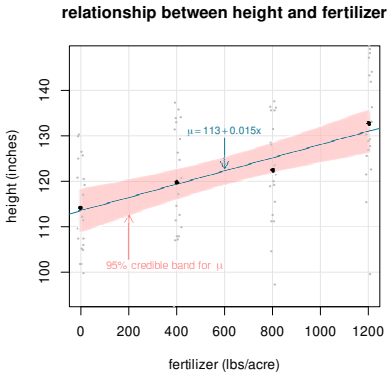


Figure 26.11: Linear regression model with 95% uncertainty band for $\mu(x)$.

Finally we add the regression line, plot points, and additional annotations:

```
abline(a=a.PE,b=b.PE,col="#127893")
points(x=jitter(corn.dat$fert,factor=0.5), y=corn.dat$h,
       pch=19,col="gray",cex=0.2)
points(x=x,y=lapply(raw.data,mean),pch=19,cex=0.6)
text(x=410,y=130.5,pos=4,cex=0.75,col="#127893",
     labels=expression(hat(mu) == 113 + 0.015 * x))
arrows(x0=600,y0=129.5,x1=600,y1=123,length=0.1,col="#127893")
text(x=70,y=101,pos=4,cex=0.8,col=rgb(1,0,0,0.4),
     labels=expression("95% credible band for " ~ mu))
arrows(x0=200,y0=103,x1=200,y1=112.5,
       length=0.1,col=rgb(1,0,0,0.4))
```
(R.26.20)

The resulting plot is shown in Fig. 26.11. Again, we emphasize that this 95% credible band for $\mu(x)$ reflects our *uncertainty* in the value of $\mu(x)$. The scatter of the data points around the line (i.e. the residuals) approximate the scatter around the true line (i.e. the errors) that is due to the noise characterized by σ.

26.5 Coefficient of Determination (R-squared)

26.5.1 Definition

While our analysis seems to have done a reasonable job of establishing the relationship between the mean height μ and fertilizer amount x, note that *most* of the variation in the h_i data is due to noise, and not to x. To quantify this, first we define the *residual sum of squares* SS_{res} as

$$SS_{\text{res}} = \sum_{i=1}^{n} \left(h_i - \hat{\mu}\left(x_i\right)\right)^2 \tag{26.13}$$

$$= \sum_{i=1}^{n} \Big(h_i - \underbrace{\left(\hat{a} + \hat{b} \cdot x_i\right)}_{\hat{\mu}(x_i)} \Big)^2 \tag{26.14}$$

Letting $\bar{h}$ denote the sample mean for all the heights h_i taken together, the *total sum of squares* is

$$SS_{\text{tot}} = \sum_{i=1}^{n} \left(h_i - \bar{h}\right)^2 \tag{26.15}$$

Note that SS_{tot} is just the sample variance of the h data, Eq. (3.1), without the usual $n-1$ factor.

The fraction of the total variance that is unexplained by the regression equation, i.e. the variation due to noise, is $SS_{\text{res}}/SS_{\text{tot}}$. It therefore follows that the fraction of the total variance that IS explained by the regression equation, i.e. due to the predictors, is

$$R^2 \equiv 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} = 1 - \frac{\sum\left(h_i - \left(\hat{a} + \hat{b}\cdot x_i\right)\right)^2}{\sum\left(h_i - \bar{h}\right)^2} \tag{26.16}$$

The quantity R^2 is known as the `coefficient of determination`. Again, it measures what fraction of the variation in the h data results from the linear regression equation itself.

For a general regression analysis with a response variable y, if $\hat{y}_i$ denotes the value of y_i predicted by the regression equation (which could involve multiple predictors), the coefficient of determination is

$$R^2 \equiv 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} = 1 - \frac{\sum\left(y_i - \hat{y}_i\right)^2}{\sum\left(y_i - \bar{y}\right)^2} \tag{26.17}$$

26.5.2 Computing R-squared

We can compute R^2 in R as

(R.26.21)
```
SSR <- sum( (corn.dat$h - model$fitted.values)^2 )
SST <- sum( (corn.dat$h - mean(corn.dat$h))^2 )
R.sq <- 1-SSR/SST
```

which yields a coefficient of determination $R^2 = 0.24$. Thus, only 24% of the variation in the h_i is directly attributable to the regression line. The other 76% of the variation is due to noise.

26.5.3 Interpretation of R-squared

It is typically said that a regression model is a "good fit" if R^2 is high, and a "poor fit" if R^2 is low. However, the words "good" and "bad" can be misleading. Let's unpack this.

If the goal of the regression is to accurately *predict individual heights* h_i, then our model does a poor job, since most of the variation the h_i is determined by noise and not by the regression line. Our coefficient of determination is low because the regression line doesn't determine the individual values of h_i very well. In order to improve the predictive ability of our model, we would need to take measurements of some of the noise factors and turn them into predictors.

Even without additional predictors, improvement in model prediction could of course also result from changing the model itself. As mentioned in Sec. 26.1.3, perhaps a straight line isn't the correct regression equation.

On the other hand, if the goal of the regression is simply to infer the relationship between mean corn height and the amount of fertilizer, then the results of this

chapter indicate the regression worked out pretty well. Because our goal isn't *prediction of individual heights*, the low value of R^2 isn't nearly as relevant. Instead, our goal is the *prediction of the average height* of plants in a field that received a certain amount of fertilizer, and for this our regression does pretty well. In fact, if we redid the entire analysis using only the mean heights in each field as our data, we would have found $R^2 = 0.94$.

What value of R^2 is then considered "good enough"? It depends upon how inherently noisy the data is going to be in a certain context. In a complicated social science situation, $R^2 = 0.5$ may be considered a good fit. In a tightly controlled physics experiment, $R^2 = 0.99$ might be considered a very poor fit. "Is R^2 high enough?" is not really a "yes/no" type of question.

26.6 Least-Squares Regression

26.6.1 The Basic Idea

Our specific Bayesian analysis in this chapter assumed two things: weakly-informed (or flat) priors, and patternless/random noise that doesn't vary with the predictor x. With these assumptions, it can be shown that our Bayesian point estimates $\hat{a}$ and $\hat{b}$ are the ones that make SS_{res}, i.e. Eq. (26.14), as small as possible. This will result in **the line that the data points "hug" more closely than any other possible line.** Instead of doing a full Bayesian analysis, we could find point estimates for the regression parameters by determining the numbers a and b that minimize the sum of the squares of the residuals r_i. This approach, valid under the two assumptions stated above, is called `least-squares fitting`.

26.6.2 Least-Squares in R

Instead of the Bayesian `stan_glm()`, the least-squares fit is found with the base-R function `glm()`:

In fact, `glm()` was a part of R long before Stan came along. The syntax of `stan_glm()` was chosen to mimic that of `glm()`. There is actually a more popularly used function called `lm()`, but this only deals with linear regression. We instead use `glm()` because it deals with a wider variety of regression models known as *generalized linear models*, which we discuss in Ch.29.

```
76  model.ls <- glm(h ~ fert, data=corn.dat)
```
(R.26.22)

which returns least-squares point-estimates

$$a = 113.5 \text{ in} \qquad b = 0.0146 \frac{\text{in}}{\text{pound/acre}} \tag{26.18}$$

that match the Bayesian point estimates. We can also extract frequentist confidence intervals for these parameters via the `confint()` function:

```
77  FCI.ls <- confint(model.ls,level=0.95)
```
(R.26.23)

Because our Bayesian credible intervals from Eqs. (26.7) and (26.8) assumed a weakly informed prior and normally distributed noise, they match the confidence intervals computed with `confint()`.

One nice feature of doing a least-squares analysis in R is that the `visreg` package makes it very easy to generate a plot with uncertainty bands by simply typing `visreg(model.ls)`. We'll add some extra annotation: the code

(R.26.24)
```
library(visreg)
visreg(model.ls,
       points=list(cex=0.1),
       main="Relation of corn height and fertilizer",
       xlab="fertilizer (lbs per acre)",
       ylab="height (inches)")
points(x=x,y=lapply(h.df,mean),pch=19)
text(x=200,y=144,pos=4,
     labels=expression(mu == 113.5 + 0.015 %*% fert))
```

produces the plot shown in Fig. 26.12. The uncertainty band, which is comprised of frequentist confidence intervals for $\mu(x)$, agrees with the uncertainty band in the much-harder-to-create Fig. 26.11.

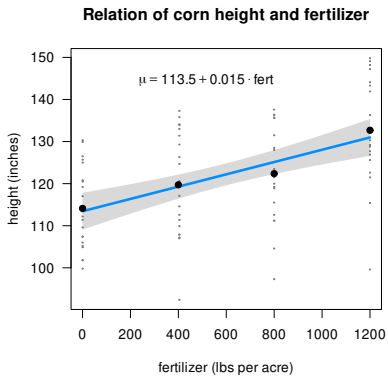


Figure 26.12: Least-squares regression model with 95% confidence level band for $\mu(x)$.

26.6.3 When You Can't Use Least-Squares

Least squares analysis is not a good choice in the presence of an informed prior probability.

Least squares also doesn't work when the noise in the data set varies with the predictor or is otherwise non-random. For example, consider Fig. 26.13. In graph (a) there is no clear pattern to the noise, and the general size of the residuals (which provide approximate estimates of the errors) doesn't seem to change much as we move along the line. This dataset would be a good candidate for least-squares regression.

In graph (b), however, the general size of the residuals clearly increases as we move to the right. When the general size of the errors (approximately estimated by the residuals) depends upon the values of the predictors, the situation is denoted as `heteroscedastic`, in contrast with the `homoscedastic` errors of graph(a). Least-squares regression will not be accurate for (b). To deal with (b), one could perform a Bayesian analysis that models the increase in the size of the noise with the predictor x, though this increases the complexity of the computation.

In graph (c), there is a clear pattern in the residuals: there is a periodic variation (or oscillation). The residuals clearly march up and down around the best-fit line. This discernible pattern means the noise can't be regarded as fully random, and least-squares won't work. Likely there is some predictor missing from our data. Accounting for this predictor (measuring it and including it in the model) might result in satisfying the requirements of least-squares analysis, though in such a case the regression line would look curvy. However, as we discuss in Ch. 28, linear regression models can in fact involve curves!

One final comment: so long as priors are weak and the noise is pattern-free (random) and independent of the predictor x, least-squares analysis does not require that the noise is normally distributed. However, if the noise is not normal, the confidence intervals computed via `confint()` are incorrect. One can compute confidence intervals for other types of probability distributions following the general methods of Sec. 21.4, though we do not pursue it here.

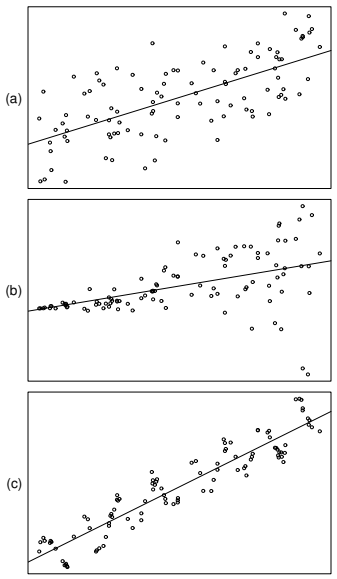


Figure 26.13: In figure (a) the residuals seem to have no pattern, and are roughly the same size for different predictor values. In figures (b) and (c) the general size of the residuals seems to depend upon the predictor value, i.e. where we are along the line. Hence, least-squares regression won't be accurate for (b) and (c).

26.7 Exercises

1. R has a built-in data frame called `Orange`. We are interested in determining the impact of age on tree circumference:
   i) Use `stan_glm()` to compute a Bayesian single-predictor linear regression model.
   ii) Make a `pairs()` plot of the resulting Bayesian model. Based on the heat-maps, which parameters seem to be related to each other? Provide explanations for these relationships or lack thereof.
   iii) Find the mean point estimates of the regression parameters.
   iv) Using the point-estimate values, on average how much circumference does the tree gain each day?
   v) Write the point-estimate regression equation and use it to estimate the circumference of the tree at 1800 days. How confident are you in this prediction? Explain.
   vi) Find 95% credible intervals for the regression parameters.
   vii) Compute the coefficient of determination for the model.
   viii) How much of the variation in circumference is due to noise? Hypothesize 5 possible sources of noise for the situation.
   ix) Make a QQ plot to evaluate whether the noise is adequately described by a normal distribution.
   x) Use `visreg()` to create a fully labeled plot containing an uncertainty band for the regression line. Use the `text()` function to print the regression equation on the plot, as in Fig. 26.12.

2. For the `mtcars` dataset we intend to evaluate the impact of `hp` on `mpg`.
   i) Use `stan_glm()` to compute a Bayesian single-predictor linear regression model.
   ii) Find the mean point estimates of the regression parameters.
   iii) Using the point-estimate values, on average how much mpg does a car lose with each additional hp?
   iv) Write the point-estimate regression equation and use it to estimate the mpg of a 400 hp car and a 20 hp car. How confident are you in these predictions? Explain.
   v) Find 95% credible intervals for the regression parameters.
   vi) Compute the coefficient of determination for the model.
   vii) How much of the variation in mpg is due to noise? In addition to the unused predictors in the dataset, list 5 possible sources of noise for the situation.
   viii) Make a QQ plot to evaluate whether the noise is adequately described by a normal distribution.
   ix) Use `visreg()` to create a fully labeled plot containing an uncertainty band for the regression line. Use the `text()` function to print the regression equation on the plot, as in Fig. 26.12.

Inspection of the `pairs()` plot of `mtcars` strongly suggests that `mpg` depends upon other predictors in the dataset. As such, to obtain a better predictive model and increase R^2, we should include more than one predictor in our regression model. We will see how to do this in the next chapter.

CHAPTER 27

Linear Regression With Multiple Predictors

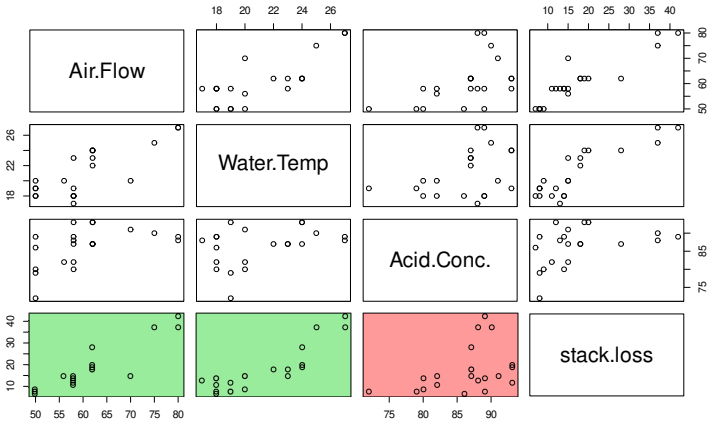
A `multi-linear regression model` is simply a linear regression model with more than one predictor variable. This introduces a few new issues which we will explain in an extended example involving the `stackloss` dataset.

27.1 Identify Predictors For `stackloss` Dataset

We will spend the rest of the chapter considering the `stackloss` data frame that is pre-loaded in R. As described in the help file, the data are from an industrial plant that converts ammonia to nitric acid. The `stack.loss` variable is a measure of the ammonia that passes through without getting processed. This `stack.loss` data is reported alongside 3 potential predictor variables: air flow, water temperature, and acid concentration. How do we decide which predictors to include in our model?

27.1.1 Examine `pairs()` Plot

In the absence of theoretical understanding of the relationships involved, one could look at the dataset with `pairs()`. The result of `pairs(stackloss)` is shown in below in Fig. 27.1.



In the red plot `Acid.Conc.` is on the x-axis and `stack.loss` is on the y-axis.

Figure 27.1: `pairs()` plot of `stackloss` data. The green plots suggest that air flow and water temperature are significant predictors for stack loss. The red plot suggests the impact of acid concentration on stack loss may be less significant.

The graphs of `stack.loss` vs. `Air.Flow` and `Water.Temp`, highlighted in green, indicate that `stack.loss` depends upon both predictors. For example, it is evident that as `Air.Flow` increases, then (generally speaking) so does `stack.loss`. The graph of `stack.loss` vs. `Acid.Conc.`, highlighted in red, shows a less clear relationship. The acid concentration may not end up being an important predictor.

Let's try to be more quantitative about this and attach some numbers to the apparent linearity between each predictor and the response.

27.1.2 Correlation Coefficients

To determine the extent to which two sets of numbers are related linearly, one can compute the correlation of their residuals. Let's explain what this means.

To start with, imagine that we have two sets of n numbers x_i and y_i. We don't assume any meaningful real-world relationship between the numbers – they could be drawn out of a hat or just made up.

Let's look at the residuals for each set of numbers. Remember from Sec. 2.3 that for a datum x_i the *residual*, which we'll call r_{xi}, is the difference between a datum and the mean:

$$r_{xi} \equiv x_i - \bar{x} \tag{27.1}$$

We make the same definition for the residuals of y_i, i.e. $r_{yj} = y_j - \bar{y}$. Now we observe the following: if x and y follow a straight line that's going uphill (i.e. with positive slope), then as we march up the line r_x and r_y should either both be positive or both be negative, i.e. they should be *correlated*, at any given point. In other words, if x increases, then so does y. Thus, if we multiply the residuals for x and y at each point, we should obtain a positive number: $r_{xi} \cdot r_{yi} > 0$. Conversely, if the line is downhill (negative slope), then as x goes up y should consistently go down, meaning the residuals have opposite sign, and $r_{xi} \cdot r_{yi} < 0$.

If, however, the points x and y don't follow a consistent linear pattern, then $r_{xi} \cdot r_{yi}$ will be positive at some points and negative at other points, meaning that on average this product of residuals is closer to zero. Thus, in order to determine whether x and y are marching uphill or downhill in sync, we want to look at the product of the residuals, $r_{xi} \cdot r_{yi}$, and add them all up:

$$\sum_{i=1}^{n} r_{xi} \cdot r_{yi} = \sum_{i=1}^{n} (x_i - \bar{x}) \cdot (y_i - \bar{y}) \tag{27.2}$$

What is the maximum possible value of this quantity, assuming the coordinates follow a perfect line? It can be shown that the maximum value is proportional to the product of the sample standard deviations [20]. If we divide Eq. (27.2) by this maximum value, we obtain an expression that can't be bigger than 1 or smaller than -1. The resulting expression is known as the `Pearson correlation coefficient`:

$$r \equiv \frac{\sum_{i=1}^{n} (x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2}} \tag{27.3}$$

It can be proven that for a least-squares linear regression model with one predictor, the correlation coefficient r (between the predictor and response data) and the coefficient of determination R^2 are related via $r^2 = R^2$.

Fig. 27.2 shows the Pearson correlation coefficient r for a variety of different plots

of x, y coordinates. When the points exactly follow a straight line, r is either $+1$ or -1. As the points become less well-described by a straight line, r approaches 0.

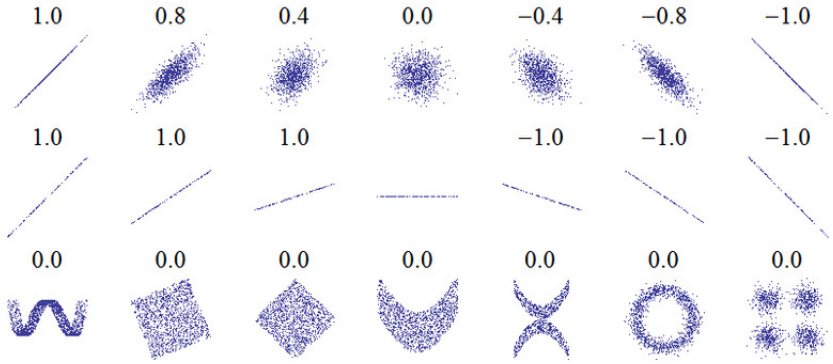


Figure 27.2: The Pearson correlation coefficient r for a variety of different x, y scatterplots, from [20]. Graphic by Wikipedia user DenisBoigelot, licensed under CC0, downloaded from Wikimedia.org.

Three important comments:

1. The correlation coefficient r does NOT quantify the degree to which two quantities are related in general – it quantifies the degree to which they are related *linearly*. The bottom row of Fig. 27.2 makes this apparent: in each of these graphs we have $r = 0$, though clearly there are noticeble *non-linear* relationships between x and y.

2. Two quantities can linearly affect each other in real-life, but you might find $r \approx 0$ just due to noise or `masking` by other predictors.

3. Even if r is a number close to ± 1, this only states that the numbers march up or down together – it doesn't imply that one actually causes the other. More on this next section.

`Masking` means that as x increases, other predictors vary in such a way that you don't see a consistent pattern in y, leading to a tiny r between x and y. If these other predictors are included in the regression model, the impact of x on y becomes apparent.

One can compute Eq. (27.3) in R with the `cor()` function. For example, to compute the correlation between `Acid.Conc.` and `stack.loss`, we type

(R.27.1)
```
cor(stackloss$Acid.Conc.,stackloss$stack.loss)
```

and get $r = 0.39$. To get all values of r between each pair of variables, type `cor(stackloss)` to get the `correlation matrix`

(R.27.2)
```
             Air.Flow Water.Temp Acid.Conc. stack.loss
Air.Flow    1.0000000  0.7818523  0.5001429  0.9196635
Water.Temp  0.7818523  1.0000000  0.3909395  0.8755044
Acid.Conc.  0.5001429  0.3909395  1.0000000  0.3998296
stack.loss  0.9196635  0.8755044  0.3998296  1.0000000
```

These numbers can also be visualized via a `correlogram`

(R.27.3)
```
library(corrgram)
corrgram(stackloss,lower.panel=panel.pie,upper.panel=panel.pts)
```

to get Fig. 27.3.

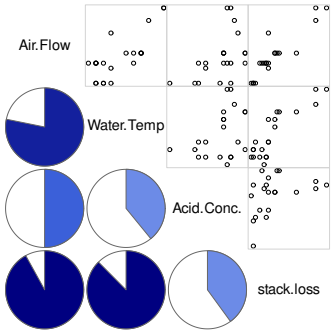


Figure 27.3: Correlation coefficient graph for `stackloss` using `corrgram()`. Bigger values of $|r|$ are indicated both by larger pies and also darker shades. Blue means positive r, and red means negative r.

With regard to `stack.loss`, the r for `Air.Flow` and `Water.Temp` are pretty high, but for `Acid.Conc.` is somewhat low. We expect both `Air.Flow` and `Water.Temp` will be important predictors in our linear model, but we're not sure about `Acid.Conc.`. We'll include all three predictors for now, and later reevaluate the inclusion of `Acid.Conc.`.

27.1.3 Be Wary of Spurious Correlations

We pause here to emphasize a simple yet very important point: just because two variables are correlated (i.e. r is close to ± 1) doesn't mean that there is any meaningful real-world relationship between them, or that one *causes* the other. Spurious (meaningless) correlations generally have two sources:

1. Confounding – The two sets of numbers x and y may not have any direct real-world relationship with each other, but there may be another variable z that influences x and y separately in a correlated direction. Such a variable is known as a `confounding variable` or `lurking variable`.
2. Pure coincidence – There are practically an infinite number of different datasets in the world, and just by chance some of them will be correlated, even if they have nothing to do with each other.

For an example of a possible confounding variable, consider a 2012 article [21] from the New England Journal of Medicine that showed a very strong correlation between a country's number of Nobel prize winners and its chocolate consumption. This led to interest in the possibility that eating chocolate could improve thinking ability. However, there is at least one obvious confounding variable here: the wealth of the society. The wealthier the society, the better the education system (on average), and hence presumably more Nobel prizes on average. But wealthy countries also have access to more luxury goods (like chocolate) compared to poorer societies. Chocolate and Nobel prizes may be causally unrelated yet correlated due to the shared impact of societal wealth.

There are other logical problems here as well. For example, no data was presented for the chocolate consumption of the actual Nobel prize winners. Perhaps none of them ate choclate. In fact, it is logically possible that chocolate *hurts* thinking ability but that this is masked by the confounding variable of societal wealth!

Many hilariously absurd examples of pure-coincidence correlations are shown at Tyler Vigen's *Spurious Correlations* website [22], including the example shown in Fig. 27.4, which has $r = 0.947$.

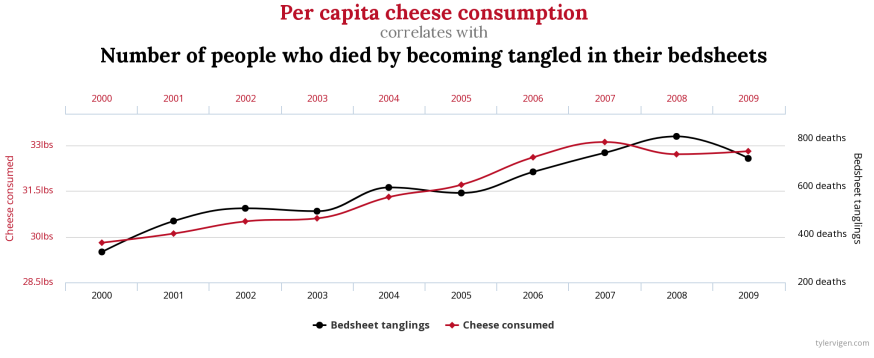


Figure 27.4: Correlation of cheese consumption and dying from getting caught in bedsheets.

Graphic by Tyler Vigen, licensed under CC BY 4.0 , downloaded from `www.tylervigen.com/spurious-correlations`.

Note that a high value of r means that two data sets form a nearly perfect straight line when plotted against *each other*. If both of these data sets (deaths and cheese consumption) are plotted against something else (e.g. time in Fig. 27.4), the graph may be very curvy. This curviness doesn't have anything to do with r. What matters for r is simply that the two data sets march in step: if one goes up, the other goes up. If one goes down, the other goes down.

Never forget one of the most important adages of statistical inference: **correlation doesn't imply causation.** It's a straightforward logical point, but human beings are pattern-seeking creatures, and we are prone to seeking causal relationships even in the absence of good evidence. In seeking to extract meaning from data, hypervigilant critical thinking is always in order!

27.1.4 Specify The Regression Model

For convenience let's denote the variables as AF, WT, AC, and SL. As in the previous chapter, we will assume that the values SL_i of the response variable are draws from a normal distribution:

$$SL_i \sim \text{normal}\Big(\mu\big(AF_i, WT_i, AC_i\big)\ ,\ \sigma\Big) \tag{27.4}$$

Here the mean of the normal population for SL_i depends upon the values of the three predictors AF_i, WT_i, and AC_i. And as before, we are assuming the standard deviation σ, characteristic of the noise, is the same for all values of the predictors. If we further assume that μ depends linearly on the 3 predictors, i.e. assume a linear regression equation, then we would have

Recalling the discussion of Sec. 26.6.3, the assumption of constant σ is referred to as *homoscedasticity*.

$$\mu = a + b_{AF} \cdot AF + b_{WT} \cdot WT + b_{AC} \cdot AC \tag{27.5}$$

Our regression model involves five parameters:

- a is the "vertical intercept" of the line, i.e. the value of the response variable SL when all of the other predictors are zero.
- b_{AF}, b_{WT}, and b_{AC} quantify how much the corresponding predictor affects the response variable. These are like the slope b from the previous chapter.
- σ is a measure of the noise, i.e. the variation in SL not due to the predictors and regression equation.

We cannot visualize our full regression line, because with 3 predictors and 1 response variable the line actually exists in 4 dimensional space! Throughout the chapter we will investigate other ways to visualize the regression.

All five of these parameters are uncertain, and hence in a Bayesian analysis are described with probability distributions.

A general multilinear homoscedastic (i.e. σ =constant) Bayesian regression model for a response variable y and p predictor variables x_j is:

$$y_i \sim \text{normal}\Big(\mu\big(x_{1i}\ ,\ x_{2i}\ ,\ \cdots\ ,\ x_{pi}\big)\ ,\ \sigma\Big) \tag{27.6}$$

where

$$\mu\big(x_{1i}\ ,\ x_{2i}\ ,\ \cdots\ ,\ x_{pi}\big) = a + \sum_{j=1}^{p} b_j x_{ji} = a + b_1 x_{1i} + b_2 x_{2i} + \cdots + b_n x_{pi} \tag{27.7}$$

Take care to note the meaning of the subscripts in Eqs. (27.6) and (27.7). The subscript i indicates which of the n data-points is being considered. Hence, y_i refers to the i^{th} value of the response variable. The subscript j indicates which of the p response variables is being considered. Thus, x_{ji}, equivalently denoted $x_{j,i}$, means the i^{th} datum-value for predictor x_j. If there were $p = 5$ different predictor variables, then x_3 would indicate the 3^{rd} of these variables, and $x_{3,9}$ would indicate the 9^{th} datum value of this predictor.

A schematic of the model specified in Eqs. (27.6), (27.7) is shown in Fig.27.5 below:

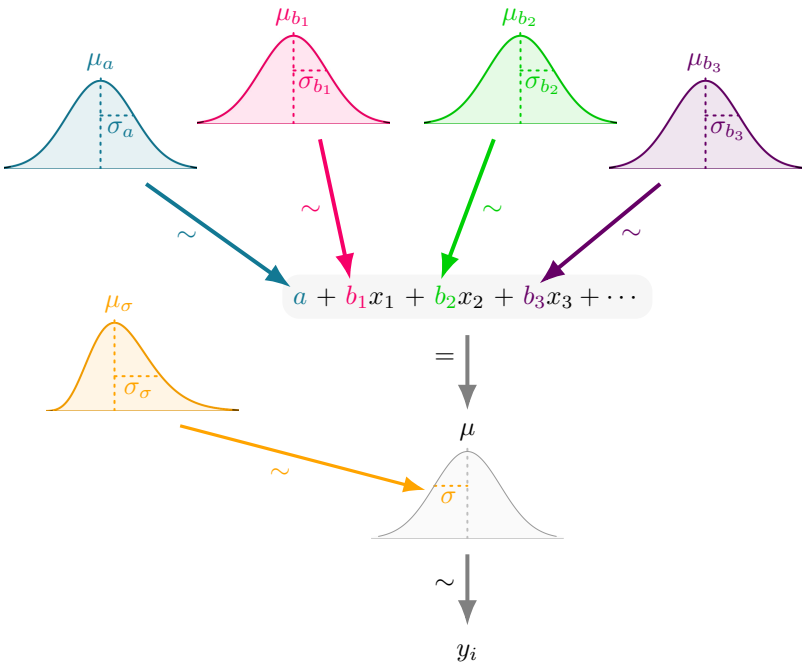


Figure 27.5: Schematic of a Bayesian multilinear regression model for response variable y and predictors x_i, with the assumption of homoscedastic errors.

σ_σ is admittedly confusing notation. As a reminder of its meaning, if we color code the expression as σ_σ then σ is our uncertainty of σ, where σ is the noise in h_i.

As in the previous chapter, we will let `stan_glm()` choose reasonable weakly-informed priors.

27.2 Compute Parameter Values in R

We relabel `stackloss` for convenience, run `stan_glm()`, and extract posterior draws:

```
D <- stackloss                                                    (R.27.4)
colnames(D) <- c("AF","WT","AC","SL")
library(rstanarm)
model <- stan_glm(SL ~ AF + WT + AC,
                  data=D,
                  iter=2e4,cores=4)
draws <- as.data.frame(model)
```

Take note of the unusual way the `+` operator is used in `stan_glm()`: the expression `SL ~AF + WT + AC` is shorthand for the regression equation Eq. (27.5). Other operators like `*` also have a special meaning inside functions like `stan_glm()`, `glm()` and `lm()`. This in turn dictates special notation is needed when specifying regression equations that are non-linear in the predictor, which we'll explore in Sec. 28.2.

We can quickly compute the 95% credible intervals and the mean point estimates for the parameters via

```
CI <- lapply(draws,function(x) quantile(x,probs=c(0.025,0.975)))  (R.27.5)
PE <- lapply(draws,mean)
```

with results given in Tab. 27.1 below. The point-estimate regression equation Eq. 27.5 for the population mean stack loss μ is thus

$$\mu = -40.0 + 0.714 \cdot AF + 1.30 \cdot WT - 0.151 \cdot AC \tag{27.8}$$

We now examine how well our regression model seems to work.

parameter	point estimate	95% CI
a	-40.0	$(-65.6, -13.9)$
b_{AF}	0.714	$(0.424, 1.00)$
b_{WT}	1.30	$(0.508, 2.09)$
b_{AC}	-0.151	$(-0.491, 0.185)$
σ	3.46	$(2.47, 5.00)$

Notice that the 95% CI for b_{AC} covers both positive and negative numbers, meaning we aren't even sure if acid concentration has a positive or negative impact on stack loss. This is more evidence that `AC` is a weak predictor in our model.

Table 27.1: Regression parameters computed by `stan_glm()` using default weakly-informed priors.

27.3 The Coefficient of Determination

We first compute R^2, i.e. how much of the variation in the data points is predicted (or "explained") by the regression line. Following (R.26.21) we type

(R.27.6)
```
SSR <- sum( (D$SL - model$fitted.values)^2 )
SST <- sum( (D$SL - mean(D$SL))^2 )
Rsq <- 1-SSR/SST
```

and get 91%. The other 9% of the variation in `SL` is attributable either to noise or to an incorrect regression model. Again, there is no R^2 threshold past which a model becomes a "good fit", but 91% indicates a reasonable amount of predictive ability for our model.

27.3.1 Adjusted Coefficient of Determination

In general, as one includes more and more predictor variables to a regression model, the higher the value of R^2. This is because each predictor has its own coefficient b, and these can be regarded as "dials" or "knobs" that can be fine-tuned and adjusted to match up with the data. This is totally okay if the predictors are independent of each other and also *logically connected* to the response variable, but when the predictors have minimal to no real-world impact on the response, they inflate R^2 and cause us to overestimate the predictive ability of the model. This inflation of R^2 due to inclusion of additional illegitimate predictors is called `overfitting`.

The problem of overfitting is even more severe in non-linear regression, where you can often find a regression curve that passes through *every single data point*, i.e. obtain an $R^2 = 1$, if enough predictors are included.

It is often unclear whether a predictor should be included in a model. In our case, from the start we weren't sure if we should be including `AC`. This is problematic, since including the predictor might result in overfitting. To reduce the impact of overfitting, it is traditional to modify the formula for R^2 in manner that *penalizes* additional predictors that slightly improve R^2 but don't really carry their own weight. The `adjusted coefficient of determination` is defined as

$$R^2_{adj} = 1 - \frac{SS_{\text{res}}/(n-p-1)}{SS_{\text{tot}}/(n-1)} = 1 - \left(1 - R^2\right)\frac{n-1}{n-p-1} \tag{27.9}$$

where n is the number of data points and p is the number of predictors. The code

(R.27.7)
```
n <- nrow(D)
p <- 3
Rsq.adj <- 1-(1-Rsq)*(n-1)/(n-p-1)
```

gives $R^2_{adj} = 0.90$. So how does this compare to a regression model *without* `AC` as a predictor?

As an exercise you should verify that omitting `AC` barely changes the values of R^2 and R^2_{adj}. While we're not sure if `AC` should be included a predictor, it doesn't seem to be *hurting* the model or contributing to overfit, so we'll tentatively keep it.

Note that `AC` might actually be an important predictor, but we're simply not seeing it due to bad luck with the data we happened to get. If there is a good theoretical basis for assuming acid concentration would impact stack loss, we should include it. If there is a good theoretical basis for assuming `AC` doesn't impact `SL`, we should omit it.

27.4 Visualizations of The Model

Because there are three predictors in our model, the regression line Eq. (27.8) can only be plotted in four dimensions. As such, we cannot directly visualize the fit. There are, however, a number of other plots we can make to visually assess the results.

27.4.1 Predicted vs. Actual Plot

Fitted value is another term used for the *predicted value*.

Given values for the predictors AF, WT, and AC, our best guess for the value of the response variable SL is the population mean given by Eq. (27.8). Let us call this the `predicted value` of SL:

For a general linear model with predictors x_j and response variable y, the predicted value is $\hat{y} = \hat{a} + \sum_{j=1}^{p} \hat{b}_j x_j$.

$$SL_{\text{predicted}} = \mu\big(AF, WT, AC\big) \tag{27.10}$$
$$= \hat{a} + \hat{b}_{AF} \cdot AF + \hat{b}_{WT} \cdot WT + \hat{b}_{AC} \cdot AC \tag{27.11}$$
$$= -40.0 + 0.714 \cdot AF + 1.30 \cdot WT - 0.151 \cdot AC \tag{27.12}$$

For our $n = 21$ data points, we can compare the actual values and the predicted values of SL. For a perfect fit where $R^2 = 1$, the actual and predicted values match. Hence, making a plot of $SL_{\text{predicted}}$ vs SL_{actual} would result in a perfect 45° line for $R^2 = 1$. The worse the fit, the more the points would deviate from this line. The code

```
plot(x=D$SL,
     y=model$fitted.values,
     xlim=c(0,45), ylim=c(0,45),
     xlab="SL measured",
     ylab="SL predicted by linear model",
     main="Predicted vs Actual SL")
abline(a=0,b=1)
```
(R.27.8)

produces Fig. 27.6. The points are in the vicinity of the 45° line, meaning it is a reasonable fit. This is consistent with our relatively high value of R^2.

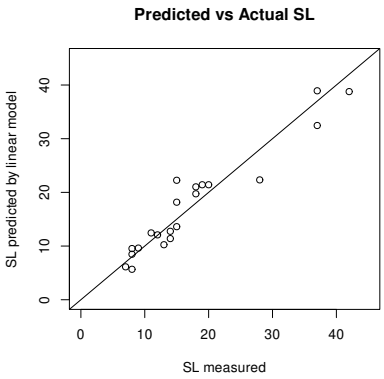


Figure 27.6: Actual vs. Predicted stack loss values.

27.4.2 Q-Q Plot

As in Sec. 26.3.2, we can assess the normality of the noise by making a Q-Q plot. The code

(R.27.9)
```
r <- model$residuals
z <- (r-mean(r))/PE$sigma
qqnorm(z,xlim=c(-2.5,2.5),ylim=c(-2.5,2.5),
       main="Q-Q Plot of Standardized Residuals")
abline(a=0,b=1)
```

gives Fig. 27.7. The points are relatively close to the 45° line, indicating that the noise is adequately described by a normal distribution.

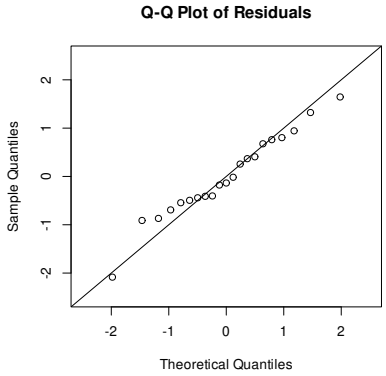


Figure 27.7: Q-Q Plot for the linear regression model.

27.4.3 Partial Residual Plots

If we wanted to visualize the relationship between SL and AF, we run into a problem: for each possible value of AF, there are an infinite number of possible values for both WT and AC. One way of addressing this is to assume that WT and AC are always at their average values, typically chosen as the median. Let us denote the resulting SL value (i.e. the predicted value of the response variable) as $\widehat{SL}_{\text{partial}}$, or the `partial predicted value`.

The regression equation becomes

$$\widehat{SL}_{\text{partial}} = \hat{a} + \hat{b}_{AF} \cdot AF + \hat{b}_{WT} \cdot WT_{0.5} + \hat{b}_{AC} \cdot AC_{0.5} \tag{27.13}$$

Notice that on the right-hand-side, the values of all symbols are specified except for AF, which is allowed to vary. Thus, AF is the only variable on the RHS. This leads to the ability to make a type of plot of SL vs. AF, and similarly for the other predictors (see Comment 27.1 for more info).

The resulting plots are quickly generated via `visreg(model)`, as shown in Fig. 27.8. Each of the three plots is known as a `conditional partial residual plot`.

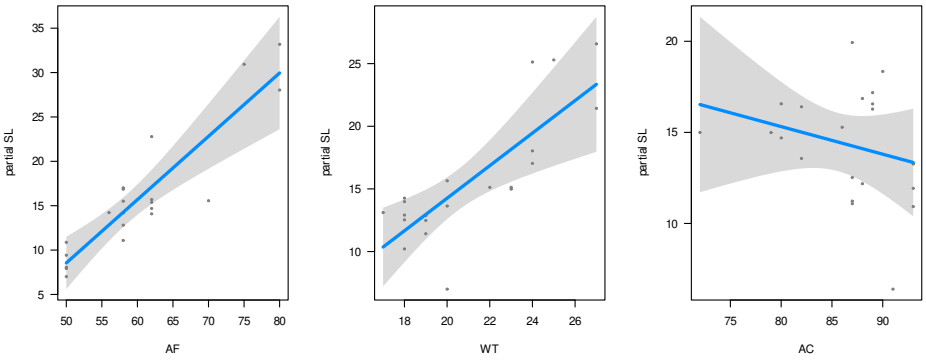


Figure 27.8: Conditional partial residual plots produced by `visreg()`.

In Fig. 27.8 the y-coordinates of the points are not the actual stack loss measurements, but instead are the values that our model says *would* have actually occurred if the other two predictors were at their median values. The best-fit lines shown have slopes matching the point estimates $\hat{b}_j$, i.e. the slope of the first graph is $\hat{b}_{AF}$, though the intercept is now different from $\hat{a}$. See Comment 27.1 for more details.

We have mentioned several times that AC is a weak predictor. Another way of seeing this is to examine the third plot in Fig. 27.8 above, where we see that within the 95% uncertainty band it's not clear whether the best-fit line should have positive or negative slope, i.e. whether AC actually increases or decreases SL.

- SL = value measured in real world
- SL_{partial} = value measured in world where other predictors are at median values
- $\widehat{SL}$ = value predicted by regression equation where predictors are at their real-world values
- $\widehat{SL}_{\text{partial}}$ = value predicted by regression equation where other predictors are at median values

Here we are assuming the residuals r_i provide estimates of the errors e_i, which is only approximately true. See Sec. 26.3.2.

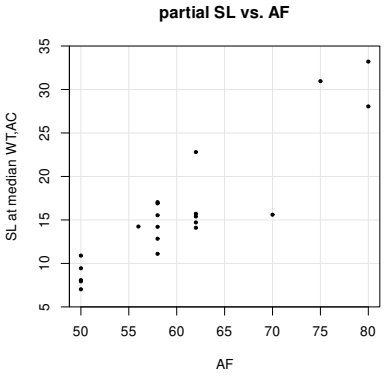


Figure 27.9: Plot of partial SL values vs. AF. These are the same points that appear in the first plot of Fig. 27.8.

Comment 27.1 – Computing Partial Values (Optional Reading)

Let's look at the SL vs. AF plot. Consider the first row of data `D[1,]`:

```
   AF WT AC SL
1 80 27 89 42
```

The predicted value of $\widehat{SL}$ using the full regression equation is

$$\widehat{SL} = \underbrace{-40}_{\hat{a}} + \underbrace{0.714}_{\hat{b}_{AF}} \cdot \underbrace{80}_{AF} + \underbrace{1.30}_{\hat{b}_{WT}} \cdot \underbrace{27}_{WT} \underbrace{-0.151}_{\hat{b}_{AC}} \cdot \underbrace{89}_{AC} = 38.75$$

If we instead use the median values $WT_{0.5}$ = `median(D$WT)` = 20 and $AC_{0.5}$ = `median(D$AC)` = 87 we have that the partial predicted value is

$$\widehat{SL}_{\text{partial}} = -40 + 0.714 \cdot 80 + 1.30 \cdot 20 - 0.151 \cdot 87 = 29.95$$

Using the median values for WT and AC dropped the predicted value from 38.75 to 29.95, i.e. lowered it by 8.8. We will therefore assume that the *measured* value for `SL` in this data point would have dropped by 8.8 if WT and AC were their median values. Instead of the actual value $SL = 42$ we would have measured

$$SL_{\text{partial}} = 42 - 8.8 = 33.2 \tag{27.14}$$

This partial value is *conditional*, in the sense that we assume SL_{partial} is what we would have measured in a world where predictors aside from AF are at their median values. In the first plot of Fig. 27.8 the y-values are SL_{partial} (for AF) and not the original SL.

Note that the predicted value $\widehat{SL}$ is 38.75, while the actual value SL was 42, meaning a residual of $r = -3.25$. Due to noise, the actual value is 3.25 higher than the predicted value. The assumption of homoscedasticity means the errors don't depend upon the predictor values, and hence the errors (and approximately the residuals) should be the same even if some of the predictors assume their median value. It then follows that $SL_{\text{partial}} = 33.2$ is 3.25 higher than the predicted partial value $\widehat{SL}_{\text{partial}} = 29.95$.

To gain some insight into how the conditional partial residual plot is constructed, let's plot the partial points without using `visreg()`. Note that recipe above: subtract the full regression residual from the predicted $\widehat{SL}_{\text{partial}}$ to get the "actual" SL_{partial}:

$$\begin{aligned} SL_{\text{partial}} &= \widehat{SL}_{\text{partial}} - r = \widehat{SL}_{\text{partial}} - \left(\widehat{SL} - SL\right) \\ &= \left(\hat{a} + \hat{b}_{AF} \cdot AF + \hat{b}_{WT} \cdot WT_{0.5} + \hat{b}_{AC} \cdot AC_{0.5}\right) \\ &\quad - \left(\hat{a} + \hat{b}_{AF} \cdot AF + \hat{b}_{WT} \cdot WT + \hat{b}_{AC} \cdot AC - SL\right) \\ &= SL + \hat{b}_{WT} \cdot \left(WT_{0.5} - WT\right) + \hat{b}_{AC} \cdot \left(AC_{0.5} - AC\right) \end{aligned}$$

A `plot()` of SL_{partial} vs. AF is shown in Fig. 27.9. The best-fit line through these points will have the slope $\hat{b}_{AF}$ from the full regression line, but the intercept will be $\hat{a} + \hat{b}_{WT} \cdot WT_{0.5} + \hat{b}_{AC} \cdot AC_{0.5}$ instead of just $\hat{a}$.

27.5 Prediction Intervals

Sometimes the goal of a regression analysis is compute estimates of regression parameters, as these can give useful information about how a predictor influences a response. Other times we wish to predict the value of a response variable for new combinations of the predictor values. As with any statistical estimate, we should also indicate our uncertainty.

Let's say we wish to know the mean stack loss SL for an air flow $AF = 44$, water temperature $WT = 22$, and acid concentration $AC = 85$. This particular combination of predictors does not occur in the dataset, and even if it did we are interested in the mean value predicted by our model, and not an actual data point (which will be on either side of this mean value due to noise).

There is a `predict()` function in R that will generate the point estimate, but it is instructive to "manually" produce the estimate. First, we compute Eq. (27.8) with the given predictor values. Continuing from (R.27.4) and (R.27.5),

The double square brackets are used to access elements of `PE` because it is a list, given how we computed it in (R.27.5). If you didn't want to deal with lists, you could also just separately compute the mean of each coefficient, i.e. `a <- mean(draws[,1])`, etc.

(R.27.10)
```
AF <- 44 ; WT <- 22 ; AC <- 85
SL.pred <- PE[[1]] + PE[[2]]*AF + PE[[3]]*WT + PE[[4]]*AC
```

which gives $SL = 7.2$. To compute a 95% credible interval, we compute a vector full of estimates for SL, each estimate coming from one of the draws (where each draw specifies a specific combination of values for the four regression parameters). We can then look at the 2.5% and 97.5% quantiles of this vector to determine the credible interval:

(R.27.11)
```
SL.pred.draws <- draws[,1] + draws[,2]*AF +
                 draws[,3]*WT + draws[,4]*AC
SL.pred.BCI <- quantile(SL.pred.draws,probs=c(0.025,0.975))
```

which gives a 95% credible interval or `prediction interval` of $(1.76, 12.5)$ to the SL value at the predictors $AF = 44$, $WT = 22$, $AC = 85$.

The meaning and computation of this interval matches that of the intervals shown in Fig. 26.11, where we computed the 95% uncertainty band for a regression line.

It is important to note that our prediction interval is not a complete specification of the uncertainty in our prediction, because our choice of regression model is also uncertain. Although quantifying model uncertainty is a difficult topic beyond the scope of this book, at the very least we should *acknowledge* model uncertainty when conducting and presenting analyses.

Reminder that this general point holds for non-regression models as well. Whenever we present the result of a statistical analysis, there is *always* an assumed statistical model in the background, and this model is always subject to undertainty.

27.6 Exercises

1. The package `MASS` has a data frame `gilgais` with a variety of soil measurements. Let's focus on three measurements at depths from 0 to 10 cm in the soil:

   - `pH00` = pH
   - `e00` = electrical conductivity
   - `c00` = chloride content

   See the help file via `help(gilgais)` for more information.

Let us suppose we are interested in examining the relationship of these three variables. We will consider a Bayesian linear regression model with response variable `e00` and predictors `pH00` and `c00`.

i) Given what you know about soils and chemistry, does it make sense *logically* that conductivity would be related to chloride content or pH? Explain.

ii) Use `corrgram()` and `pairs()` to make an educated guess about which predictor will be more important.

iii) Compute the model (using both predictors) with `stan_glm()`. Make a Q-Q plot of the normalized residuals and interpret the plot. Make partial residual plots with `visreg()` and summarize their content.

iv) What is the point-estimate regression equation?

v) What are the 95% credible intervals for the parameters?

vi) What is R^2_{adj}?

vii) Redo the analysis with just `pH00` as a predictor. What is R^2_{adj}?

viii) Redo the analysis with just `c00` as a predictor. What is R^2_{adj}?

ix) Based on your results, which predictor is more important? Does including the less-important predictor contribute to overfit?

x) Use the point-estimate regression equation to predict the average electrical conductivity for soil with `c00`=10 and `pH00`=6.

xi) Compute the 95% credible interval for your prediction in part x).

2. Consider the `mtcars` data frame. Our goal is build a linear regression model to predict gas mileage (`mpg`).

i) First perform a Bayesian linear regression for response variable `mpg` using *all* other data columns in `mtcars` as predictors. What is R^2_{adj}?

ii) Use `corrgram()` to identify what are likely the 4 highest-impact predictors, and redo the regression. What is R^2_{adj} for this model?

iii) Based on your results, does the first model (using all predictors) suffer from overfitting? Explain.

iv) Try playing with various sets of predictors to see if you can find an even higher R^2_{adj} model.

v) Use `visreg()` to make partial residual plots for your highest R^2_{adj} model. For each plot, indicate whether gas mileage is positively or negatively affected by the predictor, and offer an explanation for why that might be.

When trying out different combinations of predictors to see the impact on R^2, it can be helpful to use `lm()`, which reports R^2 and R^2_{adj} in its `summary()` output. The syntax for specifying the regression is the same between `stan_glm()`, `glm()`, and `lm()`.

27.7 Appendix - Categorical Group Comparisons Via Linear Regression

An analysis of group differences, e.g. what we did in Ch. 19, can be reformulated as a linear regression with categorical (nominal) predictor variables. It's worth noting at the outset that we aren't going to learn anything new: we'll end up with the same results as we did in Ch. 19. Recasting group comparisons as a regression problem is just a different way of organizing the analysis.

27.7.1 Contrasts

Let us reconsider the corn height data of Ch. 19. In our original treatment we assumed that each field had it's own mean μ. We labeled these μ_A, μ_B, and μ_C. We can restate the situation mathematically by saying the mean μ of a field is given by the expression

$$\mu = A \cdot \mu_A + B \cdot \mu_B + C \cdot \mu_C, \tag{27.15}$$

where $A = 1$ if we're in field A and $= 0$ otherwise, and likewise for B and C. For example, in field B we have that

$$A = 0, \quad B = 1, \quad C = 0 \tag{27.16}$$

so that Eq. (27.15) becomes

$$\begin{aligned} \mu &= 0 \cdot \mu_A + 1 \cdot \mu_B + 0 \cdot \mu_C \\ &= \mu_B \end{aligned} \tag{27.17}$$

We have essentially taken categorical or *nominal* data, i.e. the fertilizer type (A, B, or C), and converted it into quantitative binary data. Variables like A, B, and C, which can be set to 0 or 1 in order to pick out a data category (like fertilizer type), are known as `contrasts` or `dummy variables`. One can then look at an equation like Eq. (27.15) as a regression equation, where A, B, and C are the predictor variables, and μ_A, μ_B, and μ_C are the regression parameters. Table 27.2 shows the height data of Tab. 19.1 reconfigured in terms of the contrasts.

There are actually different schemes for creating contrasts. A more common scheme is to rework the math slightly so that the regression parameters are *differences* in means relative to one of the categories. First we pick a category value that we're going to compare the other categories against. Let's say A. Then, we can rewrite Eq. (27.15) as

$$\mu = \mu_A + B \cdot (\mu_B - \mu_A) + C \cdot (\mu_C - \mu_A) \tag{27.18}$$

Because A always equals 1 we just write μ_A instead of $A \cdot \mu_A$. Note that Eq. (27.18) is actually the exact same regression as Eq. (27.15), just rewritten in a different form. For example, in field B, we would have $B = 1, C = 0$ so that

$$\begin{aligned} \mu &= \mu_A + B \cdot (\mu_B - \mu_A) + C \cdot (\mu_C - \mu_A) \\ &= \mu_A + 1 \cdot (\mu_B - \mu_A) + 0 \cdot (\mu_C - \mu_A) \\ &= \mu_A + \mu_B 1.5 - \mu_A \\ &= \mu_B \end{aligned}$$

A	B	C	h
1	0	0	89.6
1	0	0	97.4
1	0	0	102.6
1	0	0	107.7
1	0	0	146.4
1	0	0	85.0
1	0	0	143.1
1	0	0	99.3
1	0	0	111.6
1	0	0	129.7
1	0	0	113.6
1	0	0	133.3
1	0	0	144.5
1	0	0	89.7
1	0	0	96.9
1	0	0	154.3
1	0	0	124.6
1	0	0	132.6
1	0	0	155.1
1	0	0	102.6
0	1	0	153.4
0	1	0	115.4
0	1	0	150.4
0	1	0	101.9
0	1	0	106.9
0	1	0	109.6
0	1	0	156.2
0	1	0	111.6
0	1	0	119.7
0	1	0	103.2
0	1	0	148.3
0	1	0	133.5
0	1	0	136.3
0	1	0	126.2
0	1	0	145.5
0	1	0	150.1
0	1	0	141.7
0	1	0	75.3
0	1	0	101.5
0	1	0	103.2
0	0	1	140.3
0	0	1	91.2
0	0	1	107.2
0	0	1	77.9
0	0	1	126.1
0	0	1	63.3
0	0	1	111.9
0	0	1	132.0
0	0	1	99.8
0	0	1	109.0
0	0	1	144.5
0	0	1	106.8
0	0	1	91.2
0	0	1	142.0
0	0	1	125.1
0	0	1	114.4
0	0	1	94.0
0	0	1	116.0
0	0	1	141.5
0	0	1	140.4

Table 27.2: Corn Height In Inches. Contrast variables reflect Eq. (27.15).

So we see that Eq. (27.18) is really the same underlying regression model as Eq. (27.15). The difference is that in the regression of Eq. (27.18), the regression parameters we'll find will be μ_A and the *differences* of the other two means from μ_A. Table 27.3 shows the corn-height data with the modified contrast scheme.

A	B	C	h
1	0	0	89.6
1	0	0	97.4
1	0	0	102.6
1	0	0	107.7
1	0	0	146.4
1	0	0	85.0
1	0	0	143.1
1	0	0	99.3
1	0	0	111.6
1	0	0	129.7
1	0	0	113.6
1	0	0	133.3
1	0	0	144.5
1	0	0	89.7
1	0	0	96.9
1	0	0	154.3
1	0	0	124.6
1	0	0	132.6
1	0	0	155.1
1	0	0	102.6
1	1	0	153.4
1	1	0	115.4
1	1	0	150.4
1	1	0	101.9
1	1	0	106.9
1	1	0	109.6
1	1	0	156.2
1	1	0	111.6
1	1	0	119.7
1	1	0	103.2
1	1	0	148.3
1	1	0	133.5
1	1	0	136.3
1	1	0	126.2
1	1	0	145.5
1	1	0	150.1
1	1	0	141.7
1	1	0	75.3
1	1	0	101.5
1	1	0	103.2
1	0	1	140.3
1	0	1	91.2
1	0	1	107.2
1	0	1	77.9
1	0	1	126.1
1	0	1	63.3
1	0	1	111.9
1	0	1	132.0
1	0	1	99.8
1	0	1	109.0
1	0	1	144.5
1	0	1	106.8
1	0	1	91.2
1	0	1	142.0
1	0	1	125.1
1	0	1	114.4
1	0	1	94.0
1	0	1	116.0
1	0	1	141.5
1	0	1	140.4

Table 27.3: Corn Height In Inches. Contrast variables reflect Eq. (27.18).

27.7.2 Specify Regression Model

The response variable is the height h of a corn plant, which we assume is statistically modeled by a normal population

$$h \sim \text{normal}\Big(\mu(B,C), \sigma\Big) \tag{27.19}$$

where the population mean μ is a linear function of the predictors B and C

$$\mu = \mu_A + (\mu_B - \mu_A) \cdot B + (\mu_C - \mu_A) \cdot C \tag{27.20}$$

meaning that there are four regression parameters in our model:

- μ_A, the population mean for field/fertilizer A, which is the "intercept" for our equation (though the predictors aren't continuous, so there isn't really a straight line one can plot)
- $\mu_B - \mu_A$, the difference in population mean between field/fertilizer B and A, which is the coefficient of the predictor variable B.
- $\mu_C - \mu_A$, the difference in population mean between field/fertilizer C and A, which is the coefficient of the predictor variable C.
- σ, the noise in the height measurements, which is assumed to be independent of the values of the predictors, i.e. the same for all fields/fertilizers.

In a Bayesian analysis, our uncertainty in the value of these parameters is expressed through probability distributions. As usual, given that we have no relevant information prior to collection of the data, we will use the weakly-informed priors chosen by `stan_glm()`.

27.7.3 Compute + Interpret Results

We first read in the data and format it into a data frame with a column for all the heights and a column with the field labels:

```
h.df <- read.csv("corn.csv")
colnames(h.df) <- c("A","B","C")
h.df <- stack(h.df)
colnames(h.df) <- c("h","fert")
```
(R.27.12)

We can now call `stan_glm()`, which will be default use the contrast scheme of Eq. (27.18) and Table 27.3:

```
library(rstanarm)
model <- stan_glm(h~fert, data=h.df, iter=2e4, cores=4)
summary(model)
```
(R.27.13)

The results are

(R.27.14)
```
Estimates:
              mean   sd     10%    50%    90%
(Intercept) 117.97   5.23 111.33 117.98 124.62
fertB         6.53   7.38  -2.87   6.56  15.92
fertC        -4.23   7.39 -13.61  -4.28   5.26
sigma        23.23   2.20  20.57  23.06  26.13
```

We interpret the mean parameter estimates as

$$\begin{aligned} \mu_A &= 118.0 \\ (\mu_B - \mu_A) &= 6.5 \\ (\mu_C - \mu_A) &= -4.2 \end{aligned}$$

which in turn implies the point estimates

$$\begin{aligned} \mu_A &= 118.0 \\ \mu_B &= 124.5 \\ \mu_C &= 113.8 \end{aligned}$$

and these agree with the point estimates of Ch. 19. All of the other calculations of Ch. 19 can be duplicated through computations on the draws of the regression parameters.

Again, there are no new results here, although calling `stan_glm()` and using its defaults is often easier than writing a full Stan model file and calling `stan()`. So why didn't we just use `stan_glm()` back in Part III? There are three reasons:

1. We were just introducing the basic ingredients that were later needed for understanding regression. In other words, we weren't ready to deal with full regression models at that point.
2. Not all statistical models can be cast in the form of regression, thus requiring the use of a Stan model file (or the equivalent in another program).
3. The `stan_glm()` function by default hides the choices of prior and likelihood, which is fine for users who know what they're doing, but is not good for those who are *learning*. Writing out a Stan file (while sometimes frustrating) is actually better for beginners, because it forces the complete specification of the statistical model, including priors and likelihoods.

CHAPTER 28

Nonlinearity in Predictors

Thus far we've glossed over something important, and it's time for full disclosure: when statisticians refer to *linear regression*, typically they are referring not to the shape of the regression line, but instead to the linearity of the regression parameters. Thus, linear regression can be used to describe some types of nonlinear relationships between predictors and a response variable.

28.1 The "Linear" in Linear Regression

28.1.1 Linearity in Algebra

A mathematical expression is linear in its variables only if they don't multiply each other, aren't raised to any exponents, and aren't embedded inside some nonlinear function. Let's say we have variables x and y. Examples of linear expressions are

When we say "aren't raised to any exponents", we really mean any exponents other than 1. Remember that $x^1 = x$ by definition.

- $x + y$
- $2x - 3y + 7$
- $x/2 + 70y - 56$

and examples of nonlinear expressions are

- $xy + 10$
- $x^2 + 3y + 10$
- x^y
- $\sin(3x) + 2y$

You may recall from algebra that an equation like $y = 3x + 2$ is considered linear, whereas an equation like $y = x^2 - 4$ is not considered linear.

28.1.2 Linearity in Predictors vs. Regression Parameters

Consider a response variable y and predictor variables x_1 and x_2. The regression equation

$$y = a_0 + a_1 \cdot x_1 + a_2 \cdot x_2 \tag{28.1}$$

is clearly linear in the predictors x_1 and x_2. This equation is also linear in the parameters a_0, a_1 and a_2. Why? Because these two parameters don't multiply each other, aren't embedded in another function, and aren't raised to an exponent. This linearity in the predictors is what makes Eq. (28.1) a linear regression equation.

When statisticians characterize a regression equation or model as linear, they are usually referring to the linearity of the regression parameters a_i, and not to the linearity of the predictors x_i.

As another example, the regression equation

$$y = a + b \cdot x + c \cdot x^2 \tag{28.2}$$

is clearly not linear in the predictor x, since it is raised to the 2nd power. This equation is, however, linear with respect to the parameters a, b, c. Why? Because none of these three parameters multiply each other, none of them are stuck inside another function like $\log(a)$ or something, and none of them are raised to exponents. Thus, even though the graph of y versus x is a curve, the regression equation of Eq. (28.2) is considered to be linear. On the other hand, regression equations like

$$y = a \cdot x^b \qquad \text{or} \qquad y = a + x^b \tag{28.3}$$

In Sec.28.4 we'll see that an equation like $y = a \cdot x^b$ can be transformed into a linear regression equation by taking a logarithm of both sides. However, least-squares analysis is no longer fully accurate due to the way noise is transformed by this operation.

are not linear in the regression parameters a and b, since b is stuck inside an exponential function, and furthermore a is multiplying this function of b in the first one. The methods of linear regression analysis simply do not work in such cases. In Ch. 29 we discuss *generalized linear models*, wherein some special types of nonlinear regression equations can be reformulated as a type of linear regression.

28.2 Polynomial Regression

The `cars` dataset that's preloaded in R contains measurements of stopping distances (how far a car goes after the brakes are applied) versus initial speed. It stands to reason that the faster a car is initially going, the farther it will travel while coming to a stop. Elementary physics suggests that the relationship between stopping distance d and initial velocity v is roughly quadratic. Thus, we may wish to consider a regression model of the form

$$\mu = a + b \cdot v + c \cdot v^2 \tag{28.4}$$

$$d \sim \text{normal}\,(\mu(v), \sigma) \tag{28.5}$$

Actually, the a term in Eq. (28.4) makes no sense, because when $v = 0$ we should have $d = 0$, so the equation would read $0 = a + b \cdot 0 + c \cdot 0^2$, which implies $a = 0$ is the only option. However, if one omits this term in a linear regression analysis, then the sum of the residuals around the best-fit curve will no longer be zero, which is problematic in several respects, so for now we will leave the term.

Assuming uninformed priors and normally distributed noise, we can use least-squares fitting via `stan_glm()` or `glm()`. Because the `glm()` function interprets symbols like `+`, `*`, `^`, etc. in special ways, we will need to use the insulating function `I()` when including expressions such as v^2, so that R realizes we mean v raised to the second power, etc. The command

```
library(rstanarm)
stop.model <- stan_glm(dist ~ speed + I(speed^2), data=cars)
```
(R.28.1)

yields the point-estimate model

$$\mu = 2.00 + 1.03v + 0.096v^2 \tag{28.6}$$

Of course we could have just assumed a regression that was only linear in the speed. Fig. 28.1 shows the data with both the curved and the straight-line regression fit. It's hard to see from the graph which is the better regression model, but keep in mind that we had theoretical background info suggesting that the curve is the more appropriate model.

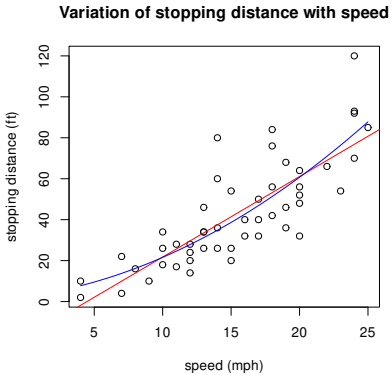


Figure 28.1: Stopping distance vs. initial speed in `cars` dataset. The least-squares fit lines are shown for regression equations linear and quadratic in the speed.

Example 28.1: Quadratic Regression

Problem: For the `airquality` dataset discussed in Sec. 8.2, use a least-squares analysis to find a polynomial regression of ozone versus temperature. Use your result to predict the ozone level at 110°F, and discuss the uncertainty in this result.

Solution: In the absence of prior knowledge regarding the relation between ozone and temperature, we look at the data for guidance:

```
with(airquality,plot(Temp,Ozone))
```

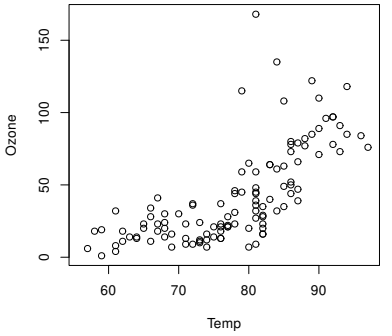


Figure 28.2: Ozone (ppb) vs. temperature (F) from `airquality` dataset.

The graph, shown in Fig. 28.2, suggests use of a quadratic polynomial. Letting O be ozone and T be temperature, our regression model is

$$O \sim \text{normal}\,(\mu(T), \sigma) \tag{28.7}$$

$$\mu(T) = a_0 + a_1 T + a_2 T^2 \tag{28.8}$$

where σ is the size of the (assumed) constant noise. Because Eq. (28.8) is linear in the coefficients a_i, we can use the methods of linear regression analysis. We compute the model with `stan_glm()`:

```
library(rstanarm)
aq.model <- stan_glm(Ozone ~ Temp  + I(Temp^2),
                     data = airquality,
                     cores = 4, iter = 5e5)
```

The `iter` parameter was determined by increasing it until the mean parameter estimates were consistent to 3 significant figures each time the R script was ran. At 4 CPU cores this was around 500,000, or 5e5.

Examining `aq.model$coefficients` gives mean values $a_0 = 217$, $a_1 = -7.22$, $a_2 = 0.063$. Thus, our point-estimate equation for the mean ozone at a given temperature is

$$\mu(T) = 217 - 7.22\,T + 0.063\,T^2 \tag{28.9}$$

We can use `visreg()` to easily visualize the fit line:

```
library(visreg)
visreg(aq.model,xlab="temperature (F)",ylab="ozone (ppb)")
```

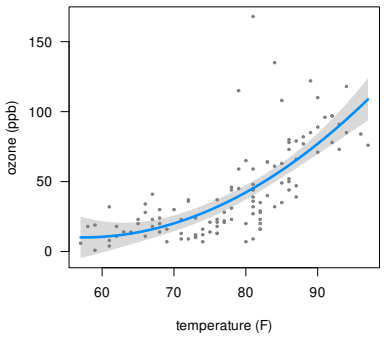


Figure 28.3: Ozone (ppb) vs. temperature (F) from `airquality` dataset, with 95% credible band for the quadratic regression equation.

with result shown in Fig.28.3. To compute our point-estimate for the expected ozone at $T = 110$°F, we simply insert $T = 110$ into Eq. (28.9):

$$\mu(110) = 217 - 7.22 \cdot 110 + 0.063 \cdot 110^2 \approx 185 \text{ ppb} \tag{28.10}$$

To compute a (just to pick a number) 90% prediction interval, we follow the steps outlined in Sec. 27.5:

```
draws <- as.data.frame(aq.model)
T.new <- 110
mu.draws <- draws[,1] +draws[,2]*T.new +draws[,3]*T.new^2
pred.int <- quantile(mu.draws,probs=c(0.05,0.95))
```

which yields the 90% prediction interval (150, 218) ppb for the ozone at 110°F. The point estimate is 184 ppb, but at the 90% credible level the ozone level could be anywhere within 34 ppb of this point estimate, due to our uncertainty in the regression parameter values. However, even if somehow there were no uncertainty in our estimate for μ, remember that an experimentally observed ozone value comes from Eq. (28.7). In other words, there is additional uncertainty (on the order of σ) in our prediction due to noise, i.e. all the possible predictors not included in our regression equation Eq. (28.8).

There are (at least) two additional sources of uncertainty in our result. One, the chosen statistical model is uncertain. Perhaps noise isn't normal, or the regression curve isn't actually quadratic in temperature, etc. Two, even if the chosen model is pretty accurate for the range of data presented, when computing ozone at $T = 110$°F we are *extrapolating* outside the temperature range in the data, and the model might be very inaccurate at such temperatures.

Note that we have listed *four* different sources of significant uncertainty in our prediction. And there are others. For example, we didn't even consider the possibility of errors in the measurement equipment or data collection. Nobody said statistical inference was an easy business!

28.3 The Peril of Overfitting

When doing scientific research, one should use complicated regression equations with great caution, primarily due to the dangers of overfitting discussed in Sec. 27.3.1. The problem is that v, v^2, v^3 etc. are essentially regarded as separate predictors by the machinery of linear regression analysis. If you add enough terms to the model, you can get a perfect fit to the data, though the resulting model is almost certainly useless for statistical inference.

As an example, consider the corn data from Table 26.1. In particular, let's just consider the mean height for each fertilizer amount.

```
h <- c(114,120,122,133)     # height in inches
x <- c(0,400,800,1200)      # amount of fertilizer in lbs/acre
```
(R.28.2)

There are four data points. The fundamental theorem of algebra implies that we can find a polynomial with 4 terms that fits the points perfectly. In our case, we consider a regression model

$$h \sim \text{normal}\,(\mu(x), \sigma) \tag{28.11}$$

$$\mu(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \tag{28.12}$$

We can find the least-squares point estimates for the parameters via

```
corn.poly <- glm(h ~ x + I(x^2) + I(x^3))
```
(R.28.3)

to obtain the point-estimate equation

$$\mu = 114 + 0.0265\,x - 0.0000423\,x^2 + 0.0000000276\,x^5 \tag{28.13}$$

which is shown overlayed on the data in Fig. 28.4. This regression curve is a perfect fit for the data, meaning the adjusted coefficient of determination is $R^2_{\text{adj}} = 1$.

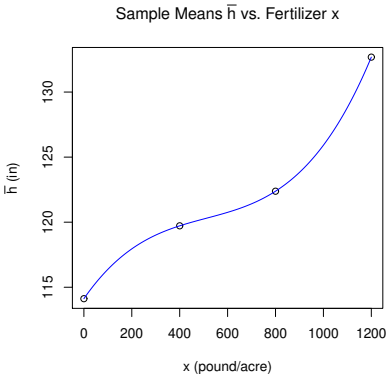


Figure 28.4: Plot of mean corn height $\bar{h}$ vs. fertilizer x. The blue curve is the cubic regression of Eq. (28.13). Fitting four data points with a cubic curve is usually ridiculous.

However, there is no reason at all to suppose that the relationship between fertilizer and height is a cubic polynomial, even over a limited range of fertilizer amount. Furthermore, if we tried to use our equation to predict corn height for a new fertilizer amount past 1200, the result would almost certainly be wrong. This is an extreme example, but it illustrates the potential problems of overfitting.

In general one should use far fewer predictors than there are data points. For example, the cubic polynomial fit shown in Fig. 28.5 seems appropriate, given that only four regression parameters are used to obtain a convincing fit of 100 data points. Unless the fit is convincing to this degree, it is generally best to avoid complicated curvy regression fits unless there is a theoretical basis for such a regression model.

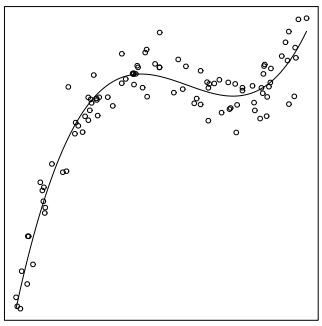


Figure 28.5: A dataset for which a cubic regression curve may be plausible.

28.4 Logarithmic Transformations

Before reading this section you may find it helpful to review the basic rules of logarithms.

The `pressure` dataset in R contains measurements of vapor pressure vs. temperature for the element mercury. We will convert the temperature readings from Celcius to Kelvin, and do some convenient renaming:

(R.28.4)
```
T <- pressure$temperature+273.15
P <- pressure$pressure
```

A plot of pressure vs. temperature is shown in Fig. 28.6. Clearly a best-fit line would be curvy. However, as you can quickly find with experimentation, no polynomial fits the data all that well. Suppose that physics theory suggests the relationship would be of the form

$$P = A \cdot e^{-B/T} \tag{28.14}$$

where A and B are positive numbers, and $e = 2.71828...$ is a mathematical constant. Eq. (28.14) is not linear in the regression parameters A and B, hence we cannot directly use the methods of linear regression analysis. However, we can transform this equation by taking a natural logarithm of both sides. Using the rule that $\ln(xy) = \ln(x) + \ln(y)$ we have

$$\ln(P) = \ln\left(A \cdot e^{-B/T}\right) \tag{28.15}$$

$$\longrightarrow \quad \ln(P) = \ln(A) + \ln\left(e^{-B/T}\right) \tag{28.16}$$

Remembering that $\ln(x)$ undoes e^x, i.e. that $\ln\left(e^x\right) = x$, we then arrive at the equation

$$\ln(P) = \ln(A) - B \cdot \frac{1}{T} \tag{28.17}$$

If we want, we can simplify the appearance of the equation a bit if we let $C = \ln(A)$, so that

$$\ln(P) = C - B \cdot \frac{1}{T} \tag{28.18}$$

$$= C - B \cdot T^{-1} \tag{28.19}$$

Regarding $y = \ln(P)$ as a new response variable, we see that we have a linear regression equation. Why? Because the parameters B and C don't multiply each

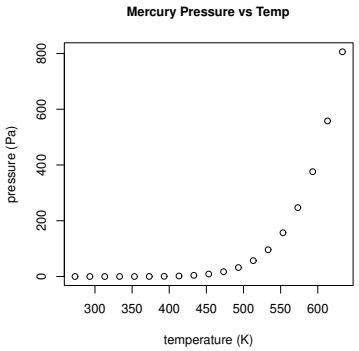


Figure 28.6: Mercury pressure vs. temperature.

other, aren't raised to an exponent, and aren't stuck inside nonlinear functions (like $\ln(x)$ or $\sin(x)$). We could then use `glm` to find the point-estimate values of these parameters.

There is one important thing we've glossed over, however. What happens to the noise when we take a logarithm? For example, if our original model is

$$P \sim \text{normal}\,(\mu, \sigma) \tag{28.20}$$

$$\mu = A \cdot e^{-B/T} \tag{28.21}$$

where σ is constant, then the noise (i.e. the errors approximated by the residuals about a best-fit curve) is normal. After taking a logarithm, the residuals will no longer be normal, instead being distorted in ways the preclude us from easily computing confidence intervals, R^2, etc. Many researchers tend to ignore this issue, but it's worth keeping in mind that something like least-squares won't be perfectly accurate after performing non-linear transformations like taking a logarithm.

Back to our equation $\ln(P) = C - B/T$. We can get approximations of the regression constants (they are approximate for reasons stated in the previous paragraph) with the R code (we could also use `stan_glm()`)

Remember that because math operations like $+, -, *, \div, \hat{}$, etc. have special meanings inside a linear model function like `glm()`, we need to insulate the expression `1/T` inside the `I()` function so that R realizes you literally mean one divided by T.

```
PT.model <- glm( log(P) ~ I(1/T) )
```
(R.28.5)

which gives $C = 18.3$ and $B = -7300$. Thus, our best fit equation is

$$\ln(P) = 18.3 - \frac{7300}{T} \tag{28.22}$$

or equivalently, since $C = \ln(A) \rightarrow A = e^C = e^{18.3}$

$$P = e^{18.3} \cdot e^{-7300/T} \tag{28.23}$$

This regression curve is plotted over the data points in Fig.28.7. The curve seems to fit the data pretty well, though the price to pay for taking the logarithm is that the computation and interpretation of things like confidence intervals or R^2 becomes murkier.

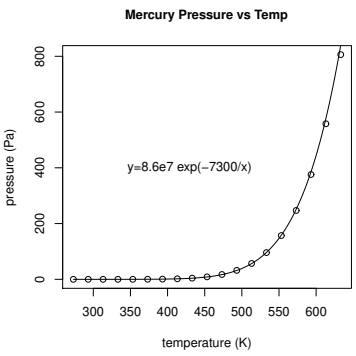


Figure 28.7: Mercury pressure vs. temperature with regression curve.

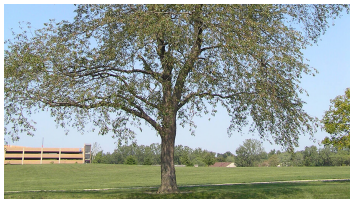


Figure 28.8: Black cherry tree.

Photo from Purdue Fort Wayne website [23].

Example 28.2: Tree Timber Volume

Problem: The `trees` dataset in R contains measurements of diameter (mistakingly labeled `Girth`), height, and timber volume for 31 black cherry trees. Find a regression model that relates timber volume to the diameter and height of the trees, and assess the model.

Solution: A tree is very complicated geometrically, but as a very crude approximation, since a good portion of the tree (the trunk and branches) are approximately circular cylinders (i.e. "a food can" shape), we could use the equation for the volume of a cylinder as a rough starting point.

A cylinder has volume $V = \pi R^2 H^1$, where R is the radius and H is the height. Expressed in terms of the diameter $D = 2R$, $V = \frac{1}{4}\pi D^2 H^1 \approx 0.79\, D^2\, H$. The volume is quadratic in the diameter and linear in the height.

Thus, as a very crude guess we might try the model

$$V \sim \text{normal}\left(\mu(D,H), \sigma\right) \tag{28.24}$$

$$\mu(D,H) = a \cdot D^b \cdot H^c \tag{28.25}$$

where for a perfect cylinder the parameters would be $a = \pi/4$, $b = 2$, $c = 1$. As it stands, Eq. (28.25) is nonlinear in the regression parameters a, b, c. However, taking a logarithm of both sides yields

$$\ln(\mu) = \ln\left(a \cdot D^b \cdot H^c\right) \tag{28.26}$$

$$= \ln(a) + \ln\left(D^b\right) + \ln\left(H^c\right) \tag{28.27}$$

$$= \ln(a) + b \cdot \ln(D) + c \cdot \ln(H) \tag{28.28}$$

which is now linear in the parameters $\ln(a), b, c$. To compute the regression parameters, first we create a copy of the data frame and convert `Girth` to feet in order to match the units of the other measurements, and then use `stan_glm()`:

```
Trees <- trees
Trees$Girth <- Trees$Girth/12        # convert in to ft
colnames(Trees) <- c("D","H","V")    # rename columns
library(rstanarm)
model <- stan_glm(log(V) ~ log(D) + log(H), data = Trees,
                  cores = 4, iter = 2e4)
```

which yields the coefficients $\ln(a) = -1.7$, $b = 1.98$, $c = 1.12$. Note that $\ln(a) = -1.7 \to a = e^{-1.7} \approx 0.18$. Thus, we arrive at a regression equation

$$\boxed{V = 0.18 D^{1.98} H^{1.12}} \tag{28.29}$$

Compared to the equation for an exact cylinder $V = 0.79 D^2 H^1$, we see the exponents match pretty closely, though the coefficient in front is about 4 times smaller. This implies the effective diameter of the trees is about 1/2 that near the base where the measurements are taken (see margin note).

Note from the help file that the measured diameter is at 4-6 feet from the ground. But of course a tree narrows towards the top. The fact that our computed a is around four times smaller than expected for a perfect cylinder suggests the rough interpretation that the effective average diameter of the overall tree is half that at 4-6 feet, since $(1/2)^2 = 1/4$, i.e. since diameter is squared in the formula for volume.

To assess the goodness of the fit, we could make a plot of the predicted vs. actual response values, following Sec. 27.4.1. Since our computation actually returns $\ln(V)$, we have to exponentiate the fitted values:

```
plot(x=Trees$V,
     y=exp(model$fitted.values),
     xlab="Measured Volume (cubic feet)",
     ylab="Predicted Volume")
abline(a=0,b=1)
```

The resulting plot is shown in Fig. 28.9. We see that the points hover close to the 45° line. This suggests that our model, while crude, nonetheless does a decent job of fitting the data. The next step would be to quantify the uncertainty in the regression parameters, for example by computing credible intervals as in previous examples.

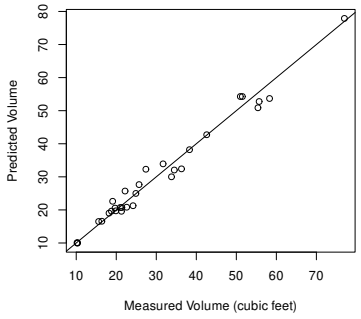


Figure 28.9: Timber volumes predicted by regression Eq. 28.29 vs. measured volume in `trees` dataset.

28.5 Interaction Terms in Multilinear Regression

While we won't delve into the topic much, it's worth mentioning that sometimes how much a predictor x_i impacts a response variable depends upon the value of another predictor x_j, and vice versa. Such predictors are said to *interact*. A simple example might be that the impact of fertilizer amount f on plant height h depends upon how much daily water w the plants receive. The regression equation for the mean plant height μ might then be

Interaction terms should indicate a logical influence of one predictor on another. For example, the fertilizer is water soluble and also requires a healthy amount of water for uptake. Or perhaps the fertilizer itself helps the plant absorb whatever water is present. Etc.

$$\mu = a + b(w) \cdot f + c(f) \cdot w \tag{28.30}$$

where we see that the regression coefficient b, indicating the impact of f, now depends upon w. Similarly for c. The simplest dependence of b on w is itself linear, i.e. $b = b_1 + b_2 w$. We then have

$$\mu = a + \big(b_1 + b_2 w\big) \cdot f + \big(c_1 + c_2 f\big) \cdot w \tag{28.31}$$

Doing a big of algebra gives

$$\mu = a + b_1 \cdot f + c_1 \cdot w + (b_2 + c_2) \cdot f \cdot w \tag{28.32}$$

Relabeling the coefficients as $a_0 = a$, $a_1 = b_1$, $a_2 = c_1$, $a_3 = b_2 + c_2$ gives

$$\mu = a_0 + a_1 f + a_2 w + a_3 f w \tag{28.33}$$

Thus, in the simplest case, the interaction between the predictors f and w manifests as an additional term that is the product of f and w. Here a_1 indicates the impact of fertilizer f independent of the value of the water w, and likewise for a_2. In regression parlance these impacts are known as `main effects`. The parameter a_3 indicates an `interaction`, meaning the impact of f that depends upon the value of w (and vice versa).

To specify Eq. (28.33) in a function like `glm()`, one simply uses the syntax `h ~ f*w`.

28.6 Exercises

1. For the `airquality` dataset, conduct a polynomial regression analysis of ozone vs. wind, using Example 28.1 as a guide. Use `visreg()` to create partial residual plots for your model, and explain the meaning/interpretation of a partial residual plot. Give the 85% prediction interval for the ozone level at a wind speed of 25 mph, and discuss the various sources of uncertainty in your result.

2. In Exercise #2 of Sec. 27.6 you attempted to build a regression model for gas mileage using the `mtcars` data frame. However, that model assumed a regression that was linear in all predictors. In this exercise we will perform a regression that is nonlinear in the predictors (though still linear in the regression parameters!) to determine a relationship between *mpg* and *hp*.

    i) Make a plot of *mpg* vs. *hp* for the data. Does it seem like a straight line or polynomial would be a good fit?

ii) Assume a regression model

$$mpg \sim \text{normal}\left(\mu(hp), \sigma\right) \tag{28.34}$$

$$\mu(hp) = a + \frac{b}{hp} \tag{28.35}$$

and use `stan_glm()` to calculate the linear model. Find point-estimates of the parameters a and b, along with 95% credible intervals for these parameters.

iii) Make a plot of *mpg* vs. *hp* and overlay the point-estimate regression curve. Next, make a predicted vs. actual plot for the model. Qualitatively speaking, how good of a fit does it seem to be? How does this "goodness of fit" relate to the noise? What other predictors do you think should/could be included in the model to improve fit?

iv) Use the model you computed in ii) to predict the mpg of a car with 140 hp, and give the corresponding 90% prediction interval. How accurate do you expect this *interpolation* to be? Explain.

v) Under the assumption that the burning of fossil fuels is bad for the environment, what recommendations would you make to consumers regarding the engine horsepower of vehicles they purchase?

CHAPTER 29

Generalized Linear Models

In this chapter we introduce the concept of the generalized linear model, or GLM, which is an interesting hybrid between linear regression (i.e. one that is linear in the regression coefficients) and non-linear regression. We motivate the idea of the GLM by first discussing an approach to regression with binomial data known as *logistic regression*. We then formalize and discuss the full concept of GLMs.

29.1 Logistic Regression

29.1.1 Bounded Parameters Are A Headache

Let's reconsider the statistical inference using discrete data discussed in the early chapters of Part III. We considered the tree disease rate p to be a constant parameter, but there are many situations involving binomial/dichotomous data for which the probability of "success" p is not a constant, instead depending upon other factors. For example, the disease rate p of a species of trees might depend upon the average temperature T near the trees. In other words, we seek a regression model relating p and T.

In the preceding chapters, we investigated regression equations for the mean μ of a normal density, and we had luck with regression equations that were linear in their parameters, e.g. $\mu = a + b \cdot x$ or $\mu = a + b \cdot x + c \cdot x^2$, etc. For our current binomial situation, we therefore might be tempted to propose a regression model such as

$$k \sim \text{binomial}\,(n, p(T)) \tag{29.1}$$

$$p(T) = a + b \cdot T \tag{29.2}$$

As before, k is the number of "successes" of our binomial experiment, and binomial $(n, p(T))$ simply means Eq. (13.1).

There is a huge problem with this statistical model, however: unlike μ, which could take on any value, the parameter p denotes a probability and hence is *bounded* between 0 and 1. Since lines and polynomials are unbounded, our model makes no sense: as T gets sufficiently large (either positive or negative), Eq. (29.2) will give values of p either less than 0 or greater than 1. While Eq. (29.2) might work for an unhelpfully narrow range of temperatures T, overall it's an inadequate statistical model.

29.1.2 What Regresion Equation Works?

We would like to have a regression equation between p and T that automatically constrains p to lie between 0 and 1 while allowing T (a continuous predictor variable) to take on any value. There are an infinite number of equations that meet this

criteria. So which one do we try out first? The simplest one, of course! Through a bunch of trial and error, people have discovered that the simplest function constraining p to the correct values is

$$p = \frac{1}{1 + e^{-a-bT}} \tag{29.3}$$

which is usually written in the equivalent form

$$p = \frac{e^{a+bT}}{1 + e^{a+bT}} \tag{29.4}$$

Eq. (29.4) keeps p in the correct range because when $a + bT$ gets very negative, $e^{a+bT} \to 0$ so that $p = 0/(1+0) = 0$ and when $aT+b$ gets very positive, $e^{a+bT} \to \infty$ so that $p = \infty/(1 + \infty) = 1$. The relationship between p and $a + bT$ is shown in Fig. 29.1. Eq. (29.4) is an example of a logistic regression equation. No doubt Eq. (29.4) is an ad-hoc and arbitrary equation, but it keeps p in the proper range, and sometimes it provides a good approximate description of things.

All the "simple-looking" alternatives involve graphs with sudden jumps or stops, which makes calculus hard to do, or involve periodic functions like $f(x) = \frac{1}{2} + \frac{1}{2}\sin x$ that go up and down repeatedly and don't make sense here.

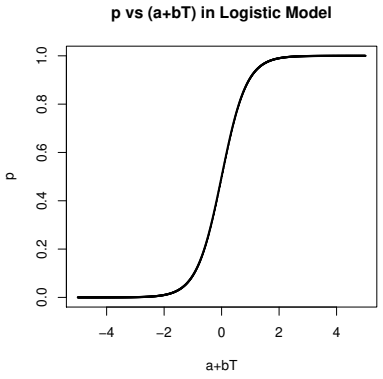


Figure 29.1: Logistic Relationship between p and $a+bT$ as given in Eq. (29.4).

29.1.3 The Definition of Logistic Regression

Consider a response variable that is the "number of successes" k in n binomial trials with success probability p. Further assume that p depends upon some number of predictors x_i. Then

$$k \sim \text{binomial}\,(n, p) \tag{29.5}$$

$$p = \frac{e^{a_0+a_1x_1+a_2x_2+\ldots}}{1 + e^{a_0+a_1x_1+a_2x_2+\ldots}} \tag{29.6}$$

is known as a `logistic regression model` with regression parameters a_i. In the previous section we labeled the parameters a, b instead of a_0, a_1. As usual, we just pick whatever notation seems convenient at the time. Returning to Eq. (29.4), a little algebra shows that

$$\frac{p}{1-p} = e^{a+bT},$$

and taking a natural logarithm of both sides gives

$$\ln\left(\frac{p}{1-p}\right) = a + bT \tag{29.7}$$

where $p/(1-p)$ is known as the `odds`, and the logarithm of the odds is known as the `logit`. In other words, *in logistic regression you don't model p as a linear function of the regression parameters, but instead you model the logit of p as a linear function of the regression parameters.*

What's important here is that the exponent $a_0 + a_1x_1 + a_2x_2 + \ldots$ is linear in the regression parameters a_i. It is not critical that the exponent is linear in the predictors. Thus, $a_0 + a_1x_1^2 + a_2/x_2$ would also work for logistic regression. But if the exponent were $a_0 + a_1x_1^{a_2} + a_3 \cdot x_2$ then it's no longer logistic regression.

This last point highlights the central idea of generalized linear models - parameters (like p or μ) of probability distributions must be mathematically transformed in some way in order to become linear in the predictors. This mathematical transformation is known as the `link function`. For logistic regression, the link function is the logit() function. As stated earlier, it is logit(p) that is linear in the regression parameters.

We'll discuss this in more detail in Sec. 29.2. But for now, let's make things more concrete by looking at an example.

29.1.4 A Logistic Regression Example

Suppose that in each of 10 different forests 100 trees were tested for disease. Table 29.1 shows the average annual temperature T and the number of diseased trees k measured in each forest. Let us denote the # of infected trees by k. We are assuming the data is modeled with the likelihood

$$k \sim \text{binomial}\,(100, p) \tag{29.8}$$

where the disease rate p depends upon the average ambient temperature T according to the logistic regression equation

$$\ln\left(\frac{p}{1-p}\right) = a + b \cdot T \quad \Longleftrightarrow \quad p = \frac{e^{a+b\cdot T}}{1+e^{a+b\cdot T}} \tag{29.9}$$

We can use `stan_glm()` to evaluate this model if we specify the appropriate link function, in our case the logit. Because our data is already aggregated for each forest (i.e. rather than 1000 yes/no data points, we have 10 data points with values between 0 and 100 indicating the number of trees that tested positive), we'll need to collect the positives and negatives into a single unit for R to understand the data:

temp (°F)	# diseased
34.4	9
62.3	87
67.6	86
47.4	36
45.6	26
51.6	46
70.5	92
52.9	48
74.1	97
61.0	64

Table 29.1: Disease and temperature data for 10 forests. In each forest 100 trees were tested. Temperatures are annual averages.

(R.29.1)
```
temp <- c(34.4,62.3,67.6,47.4,45.6,51.6,70.5,52.9,74.1,61.0)
positives <- c(9,87,86,36,26,46,92,48,97,64)
negatives <- 100 - positives
test.results <- cbind(positives,negatives)
tree.data <- data.frame(test.results=test.results,temp=temp)
```

Note that if our data points were just a bunch of 0s and 1s (with the corresponding temperatures), we wouldn't need the special formatting. See Ex. 29.1 for an example using raw binomial data (0s and 1s).

We now evaluate the model with `stan_glm()` by specifying that the data are modeled as draws from a binomial dsitribution, with a logit link function for the success probability p:

(R.29.2)
```
library(rstanarm)
model <- stan_glm(test.results ~ temp,
                  data = tree.data,
                  family = binomial(link="logit"),
                  cores = 4, iter = 5e4)
draws <- as.data.frame(model)
a.draws <- draws[,1]
b.draws <- draws[,2]
```

You now know the meaning of `glm` in `stan_glm()`. But if this function is for GLMs like a logistic model, why were we able to use it in previous chapters when doing simple linear regression? Because such simple linear regressions are just a special type of GLM with no link function (or more precisely, the link function is "keep the parameter as it is") and with response variables modeled as draws from a normal (i.e. gaussian) distribution. If you don't explicitly set the `family` parameter, it defaults to a value `gaussian(link="identity")`, i.e. the simple linear regression models of previous chapters.

We can now find point estimates and credible intervals as in previous examples. The results are

$$a: \qquad \hat{a} = -7.09 \qquad BCI_{0.95} = (-8.05, -6.17) \tag{29.10}$$

$$b: \qquad \hat{b} = 0.134 \qquad BCI_{0.95} = (0.118, 0.152) \tag{29.11}$$

The point-estimate regression equation is thus

$$\ln\left(\frac{p}{1-p}\right) = \hat{a} + \hat{b} \cdot T \tag{29.12}$$

$$= -7.1 + 0.13\,T \tag{29.13}$$

or equivalently

$$p = \frac{e^{a+bT}}{1+e^{a+bT}} = \frac{e^{-7.1+0.13\,T}}{1+e^{-7.1+0.13\,T}} \tag{29.14}$$

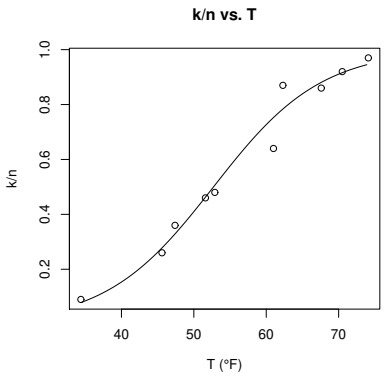


Figure 29.2: Plot of k/n vs T. The solid line is the regression equation Eq. (29.14).

veg	slope (°)
1	10
0	55
0	57
1	18
1	6
0	53
0	43
0	51
0	61
0	54
1	3
1	18
1	24
0	25
1	11
0	54
1	20
0	59
1	21
0	43
0	69
0	53
1	18
0	15
1	1
1	23
1	10
0	49
1	5
0	57

Table 29.2: Presence of a particular species of vegetation (1=yes,0=no) at coastal locations versus terrain slope.

Note that because the response variable data was already raw binomial data (0s and 1s), we didn't need to do the kind of data reformatting seen in (R.29.1).

The posterior distributions for logistic regression parameters are typically asymmetric, so taking the mean vs. the median of the draws will give slightly different point estimates for the regression parameters. In fact, `model$coefficients` returns the median value, whereas `summary(model)` returns mean values. The difference is usually small. Just be sure to indicate which one you used.

Since `positives/100` is the observed disease rate in each forest, we can plot this versus temperature and then overlay the regression line Eq. (29.14) to get a visual of the fit. The result is shown in Fig. 29.2, and seems to indicate that the fit is pretty reasonable.

Example 29.1: Modeling Species Presence

Problem: A researcher measures the ground slope (steepness) in 30 different cells (25m x 25m sample areas) of a coastal region, and further observes whether or not a specific species of vegetation is present in the cells. The raw data are shown in Table 29.2. Perform a regression analysis to determine the general impact of slope on the presence of the vegetation. Use your model to predict the probability of finding this vegetation in an area with a slope of 28°.

Solution: The response variable is binomial (vegetation is either present or not), suggesting logistic regression. Let V (0 or 1) be the presence of vegetation in a chosen sample cell, S the cell slope in degrees, and p the binomial probability of vegetation being present. We try the model

$$V \sim \text{binomial}(1, p) \tag{29.15}$$

$$\text{logit}(p) = \ln\left(\frac{p}{1-p}\right) = a + b \cdot S \tag{29.16}$$

To evaluate this model, we first store the slope data into a vector `S`, and the presence of vegetation into a vector `V`. We then form a data frame, call `stan_glm()`, and extract the parameter estimates:

```
S <- c(10,55,57,18,6,53,43,51,61,54,3,18,24,25,11,54,20,59,
       21,43,69,53,18,15,1,23,10,49,5,57)
V <- c(1,0,0,1,1,0,0,0,0,0,1,1,1,0,1,0,1,0,1,0,0,0,1,0,1,1,
       1,0,1,0)
veg <- data.frame(V=V,S=S)
library(rstanarm)
model <- stan_glm(V ~ S, data=veg,
                  family=binomial(link="logit"),
                  cores=4, iter=2e4)
draws <- as.data.frame(model)
a.draws <- draws[,1]
b.draws <- draws[,2]
```

Using `mean()` and `quantile()` on the parameter draws we find

$$a: \quad \hat{a} = 5.0 \qquad BCI_{0.95} = (2.4, 8.3) \tag{29.17}$$

$$b: \quad \hat{b} = -0.18 \qquad BCI_{0.95} = (-0.30, -0.091) \tag{29.18}$$

resulting in a point-estimate regression equation $\text{logit}(p) = 5 - 0.18 \cdot S$ or equivalently

$$p = \frac{e^{5-0.18 \cdot S}}{1 + e^{5-0.18 \cdot S}} \tag{29.19}$$

Plotting the data and overlaying Eq. (29.19) results in Fig. 29.3. It is clear

that slope has a negative impact on vegetation, i.e. the higher the slope the less likely to find this species of vegetation.

To *predict* whether vegetation will be present in a location with slope 28°, we use Eq. (29.19) with $S = 28$ to find $p = 0.5$. Thus, there is a 50% probability vegetation will be present. The probabilistic nature of our answer is of course due to noise (variation due to all the factors not in our model).

With more data we could include other predictors in our model. For example, if we had altitude measurements A for each cell, we could analyze the model $logit(p) = a + b \cdot S + c \cdot A$, call `stan_glm(V~S+A,...)` and proceed as before.

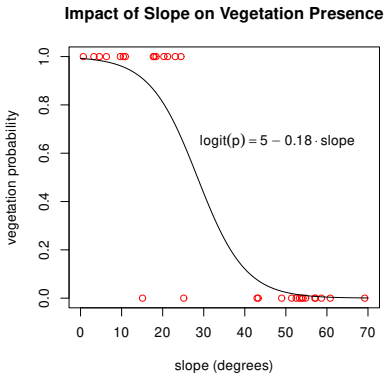


Figure 29.3: Logistic Relationship between the presence of a plant species and ground slope. While the binomial data points cannot fall on the regression curve itself, at low slope both the estimated p is large and the data are more likely to be 1, and vice-versa, suggesting a good fit.

29.2 Generalized Linear Models

29.2.1 Definition

In ordinary linear regression, we have that some probability distribution parameter (say the mean μ of a normal density) equals a linear function of the regression coefficients, e.g.

$$\mu = a + bx + cy + \dots \qquad \text{or} \qquad p = a + bx + cy + c \dots \tag{29.20}$$

In a `generalized linear model`, or GLM, it's only after you put this parameter inside another mathematical function g, called a `link function`, that it equals such a linear function of the regression coefficients.

In general, if θ is the parameter of a probability distribution from which response data is drawn, then a generalized linear model assumes

$$g(\theta) = a_0 + a_1 \cdot x_1 + a_2 \cdot x_2 + \cdots \tag{29.21}$$

where g is some link function and $x_1, x_2, \cdots$ are the predictor variables.

As with any linear model, what makes a GLM linear is that the RHS of Eq. 29.21 is linear in the regression coefficients a_i. Nonlinearity with respect to the predictors x_i is acceptable.

29.2.2 Examples of GLMs We Have Studied Already

In Sec. 29.1 we examined the important example of logistic regression. In regression Eq. (29.7) it is only after we put p inside the box $\square$ of the link function

$$g(\square) = \ln\left(\frac{\square}{1-\square}\right) = \text{logit}(\square) \tag{29.22}$$

that it equals $a + bT$, i.e.

$$g(p) = a + bT \tag{29.23}$$

As another example, in prior chapters when analyzing simple linear regression with the parameter μ of a normal distribution, we were considering the simple case of a GLM with link function $g(\mu) = \mu$, i.e. μ was already linear in the regression coefficients.

When a link function leaves the input as it is, i.e. when $g(x) = x$, it is called the *identity* link function, since the output of the function is identical to the input.

Remember that in addition to a link function, we need to specify the probability density from which the data is assumed to be drawn. In the case of simple linear regression for a response y against predictors x_i, the model often takes the form

$$y \sim \text{normal}\,(\mu,\sigma) \tag{29.24}$$

$$\sigma = \text{constant} \tag{29.25}$$

$$g(\mu) = \mu = a_0 + a_1 x_1 + a_2 x_2 + \cdots \tag{29.26}$$

which requires the (default) specification `family=gaussian(link="identity")` when calling `glm()` or `stan_glm()`. On the other hand, a logistic regression model usually takes the form

$$y \sim \text{binomial}\,(n,p) \tag{29.27}$$

$$g(p) = \text{logit}(p) = \ln\left(\frac{p}{1-p}\right) = a_0 + a_1 x_1 + a_2 x_2 + \cdots \tag{29.28}$$

which requires the specification `family=binomial(link="logit")`.

Other model specifications are possible. Because there are an infinite number of mathematical possibilities for both link functions $g(\theta)$ and probability distributions, there are infinitely many types of GLMs. In practice a small number of different GLMs are used. To see the ones programmed into R, type `help("family")`. Listed along each probability distribution is the default choice of link function that (for reasons that vary) usually work best with that type of distribution.

29.2.3 Poisson Regression

As an example of a distribution that is neither normal nor binomial, consider again the Poisson distribution discussed in Sec. 13.5. We follow the notation of that section in letting k be the number (counts) of occurrences / instances of events or objects in some region of time or space, while λ is the average counts. We now consider the dependence of λ on some number of predictor variables x_i. So what does our regression model look like?

To begin with, the data are being modeled as draws from a poisson distribution:

$$k \sim \text{poisson}\,(\lambda) \tag{29.29}$$

where the right-hand side is shorthand for Eq. 13.4. We next need to specify a link function. The typical choice is a logarithm, for reasons discussed in the Optional Reading below.

Comment 29.1 – The Log Link Function (Optional Reading)

For the poisson distribution, what would make sense as a link function for the parameter λ? In situations where the predictors tend to have an additive effect on λ, an identity link $g(\lambda) = \lambda$ may be appropriate. Remember, however, that λ represents an average count, and hence cannot be negative. Since an identity link $g(\lambda) = \lambda = a_0 + a_1 x_1 + a_2 x_2 + \cdots$ can be both positive or negative, it's often not the best choice.

We saw a reminiscent difficulty with the identity link $g(p) = p$ in binomial regression, since we needed to keep p bounded between 0 and 1. For a poisson distribution, we need λ to bounded in a different way, as a positive number.

As mentioned in Sec. 3.1, forcing things to be positive with the absolute value function causes mathematical problems. Back in that section we introduced the idea of squaring something to guarantee a positive result, but it doesn't

really work here: letting $\lambda = (a_0 + a_1x_1 + a_2x_2 + \cdots)^2$ results in an expression that is nonlinear in both the regression coefficients and the predictors, which is a mathematical mess. We could try taking a square root of both sides to get a link function $g(\lambda) = \sqrt{\lambda} = a_0 + a_1x_1 + a_2x_2 + \cdots$, but we're back to the original problem since $\sqrt{\lambda}$ also can't be negative.

The most common solution is to consider the relationship

$$\ln(\lambda) = a_0 + a_1x_1 + a_2x_2 + \cdots \qquad \longleftrightarrow \qquad \lambda = e^{a_0+a_1x_1+a_2x_2+\cdots} \tag{29.30}$$

since ln(positive #) can be both positive and negative, while e^{anything} is always positive. Thus, in Poisson regression the link function is usually chosen to be logarithmic, i.e. $g(\lambda) = \ln(\lambda)$.

The typical specification of a Poisson regression model is then

$$k \sim \text{poisson}(\lambda) \tag{29.31}$$
$$g(\lambda) = \ln(\lambda) = a_0 + a_1x_1 + a_2x_2 + \cdots \tag{29.32}$$

Analysis of the model with `stan_glm()` or `glm()` proceeds as before, except one should specify `family=poisson(link="log")`.

29.3 Exercises

1. Write R code to create the plot shown in Fig. 29.3.

2. McDonald (1985)[1] counted allele frequencies in the crustacean *Megalorchestia californiana*. The number of crabs with the Mpi^{90} allele, and the number that instead have the Mpi^{100} allele, are shown for crabs examined at 8 different locations, as shown in the Table below:

Location	Latitude	Mpi^{90}	Mpi^{100}	$pMpi^{100}$
Port Townsend, WA	48.1	47	139	0.748
Neskowin, OR	45.2	177	241	0.577
Siuslaw R., OR	44	1087	1183	0.521
Umpqua R., OR	43.7	187	175	0.483
Coos Bay, OR	43.5	397	671	0.628
San Francisco, CA	37.8	40	14	0.259
Carmel, CA	36.6	39	17	0.304
Santa Barbara, CA	34.3	30	0	0

   In this problem we will be modeling the variation of $pMpi^{100}$ (the percentage of crabs with the Mpi^{100} allele) with latitude as a logistic regression.

   i) Perform a Bayesian logistic regression analysis to see how $pMpi^{100}$ is influenced by latitude. Be sure to fully specify the statistical model, and report both point-estimates and uncertainty bands for all regression parameters.

[1] McDonald, J.H. 1985. *Size-related and geographic variation at two enzyme loci in Megalorchestia californiana (Amphipoda: Talitridae)*. Heredity 54: 359-366. This problem is adapted from McDonald, J.H. 2014. Handbook of Biological Statistics (3rd ed.). Sparky House Publishing, Baltimore, Maryland.

ii) Make a plot of pMpi^{100} vs. latitude data, and overlay the point-estimate regression curve. Assess the fit.

iii) Offer your scientific assessment of the impact of latitude on pMpi^{100} in the crabs. Feel free to perform additional calculations to support your assessment.

3. A researcher is seeking to understand things that impact the presence of a certain species of bird. As a preliminary study, they randomly select 30 regions (of equal area) and makes counts of a specific bird species seen during a one hour window, from noon to 1pm. They also measure the percent of foliage cover provided by Juniper trees in each region. The raw data is shown in Table 29.3. The goal of the preliminary study is to understand the impact of the Juniper cover on bird counts.

% cover	bird count
20	1
40	2
39	0
40	2
49	11
41	5
15	4
24	3
42	17
36	2
43	5
37	5
26	2
52	22
27	1
48	9
26	1
26	3
22	1
24	1
28	3
27	2
21	2
17	0
24	3
47	9
36	4
52	5
48	5
17	1

Table 29.3: Counts of birds (per region at the noon hour) vs. percent Juniper cover.

i) Formulate a regression model relating bird counts B to % juniper cover J. Justify the appropriateness of your chosen model type.

ii) Evaluate your model with `stan_glm()`, reporting both point-estimates and uncertainty bands for all regression parameters.

iii) Make a plot of the data, and overlay the best-fit regression equation from your model. Assess the fit.

iv) What do you conclude about the relationship between Juniper cover and bird counts? Does this relationship make sense? Offer possible explanations for this relationship.

v) No matter what values we choose for the regression parameters, and no matter how many data points (J,B) are collected the data points will not all lie on the regression curve from iii). Why is this? Use your answer to explain how our researcher might improve their research study moving ahead.

Bibliography

[1] Richard McElreath. *Statistical Rethinking: A Bayesian Course with Examples in R and STAN*. Second. Chapman and Hall/CRC, 2020. ISBN: 9780367139919.

[2] Jerold H. Zar. *Biostatistical Analysis*. Fourth. Pearson, 1998. ISBN: 9780130815422.

[3] Howard Wainer. *Visual Revelations: Graphical Tales of Fate and Deception From Napolean Bonaparte to Ross Perot*. First. Psychology Press, 2000. ISBN: 9780805838787.

[4] Stephanie Glen. *Misleading Graphs: Real Life Examples*. URL: https://www.statisticshowto.com/probability-and-statistics/descriptive-statistics/misleading-graphs/. accessed Aug. 16, 2022.

[5] Sam Savage and Jeff Danziger. "The Flaw of Averages". In: *The San Jose Mercury News* (Oct. 2000). Oct. 8th Sunday Edition.

[6] E.T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003. ISBN: 9780521592710.

[7] Robert I. Kabacoff. *Importing Data*. URL: https://www.statmethods.net/input/importingdata.html. accessed Aug. 20, 2022.

[8] Rod Pierce. *Combinations and Permutations*. URL: https://www.mathsisfun.com/combinatorics/combinations-permutations.html. accessed Aug. 21, 2022.

[9] *RStan: the R interface to Stan*. URL: https://mc-stan.org/users/interfaces/rstan.html. accessed Aug. 21, 2022.

[10] Ronald L. Wasserstein and Nicole A. Lazar. "The ASA Statement on p-Values: Context, Process, and Purpose". In: 70 (2 2016). URL: https://www.tandfonline.com/doi/full/10.1080/00031305.2016.1154108.

[11] Wikipedia. *Welch's t-test*. URL: https://en.wikipedia.org/wiki/Welch's_t-test. accessed Aug. 27, 2022.

[12] Edwin T. Jaynes and Oscar Kempthorne. "Confidence Intervals vs Bayesian Intervals". In: 1976. URL: https://bayes.wustl.edu/etj/articles/confidence.pdf.

[13] John P.A. Ioannidis. "Why Most Published Research Findings Are False". In: *PLoS Medicine* 2 (8 2005). URL: https://doi.org/10.1371/journal.pmed.0020124.

[14] Robert I. Kabacoff. *Power Analysis*. URL: https://www.statmethods.net/stats/power.html. accessed Aug. 27, 2022.

[15] Wikipedia. *Student's t-test*. URL: https://en.wikipedia.org/wiki/Student%27s_t-test#Equal_variances. accessed Aug. 27, 2022.

[16] Marek Fisz. *Probability Theory and Mathematical Statistics*. Third. John Wiley & Sons, 1963. ISBN: 0-89874-179-3.

[17] *Regression Analysis*. URL: https://en.wikipedia.org/wiki/Regression_analysis. accessed Sept. 3, 2022.

[18] Robert I. Kabacoff. *Reshaping Data*. URL: https://www.statmethods.net/management/reshape.html. accessed Aug. 27, 2022.

[19] *Leverage (statistics)*. URL: https://en.wikipedia.org/wiki/Leverage_(statistics). accessed Sept. 7, 2022.

[20] *Pearson correlation coefficient*. URL: https://en.wikipedia.org/wiki/Pearson_correlation_coefficient. accessed Sept. 3, 2022.

[21] Franz H. Messerli. "Chocolate Consumption, Cognitive Function, and Nobel Laureates". In: *The New England Journal of Medicine* 367 (2012), pp. 1562–1564.

[22] Tyler Vigen. *Spurious Correlations*. URL: https://www.tylervigen.com/spurious-correlations. accessed Aug. 16, 2022.

[23] *Native Trees of Indiana River Walk - Black Cherry*.

Index

.csv file, 64
    `read.csv()`, 64
    `write.csv()`, 64

`abline()`, 41, 75, 82
adjusted coefficient of determination, 269
adjusted R^2, 269
`aggregate()`, 255
aggregating data, 71
    `aggregate()`, 71, 76
α, *see* significance level
alternate hypothesis, 229
ANOVA, *see* NHST
`aov( )`, 242
`as.character()`, 52
`as.numeric()`, 52, 53
average, 25
    mean, 27
    `mean()`, 29
    median, 26
    `median()`, 26
    mode, 25

bar plot, 76
    `barplot()`, 76
Bayes' Theorem, 99, 139
Bayesian credible interval, *see* credible interval
Bayesian updating, 104
BCI, *see* credible interval
Behrens-Fisher density, 206
bell curve, *see* normal density
Bessel's correction, 34
beta density, 153
binning data, 17
binomial
    `dbinom( )`, 114, 141
    distribution, 113
    event, 107
    experiment, 107
    `pbinom()`, 116
    probability, 107
boxplot, 41, 75, 159
    `boxplot()`, 41
`boxplot()`, 159

`c( )`, 17
categorical data, 16, 51
`cbind()`, 63, 291
changing units, 49
chi-squared density, 130
    definition, 130
    degrees of freedom, 130
    `pchisq( )`, 131
    sample variance density, 194
χ^2 density, *see* chi-squared density
coefficient of determination, 258
Cohen's d, 229
column, 17, 58
compound hypothesis, 209
conditional probability, 86
confidence interval, 199
    Behrens-Fisher density, 206
    computing with `t.test()`, 203
    constructing, 202
    definition, 200
    differences of group means, 206
    for μ, 202
    for σ, 205
    interpretation, 201
    regression parameters, 260
    vs credible interval, 201
    Welch's t, 206
`confint()`, 260
confounding variable, 266
continuous data, 16
contrasts, 275
`cor( )`, 266
correlation coefficient
    `cor( )`, 266
correlogram, 265
`corrgram()`, 265
credible interval, 143, 150, 154
    vs confidence interval, 201
cumulative probability distribution, 116
`curve()`, 152, 153

data, 15
    binning data, 17
    changing units, 49
    continuous data, 16
    data types, 15, 16
    dichotomous data, 16

discrete data, 16
index, 20
nominal data, 16
ordinal, 16
paired data, 18
qualitative data, 15
quantitative data, 15
time series, 18
transforming, 49
data frame, 57
adding rows and columns, 63
cbind(), 63
column, 58
creating, 62
data.frame(), 62, 63
head(), 75
index notation, 59
order(), 65
rbind(), 64
renaming rows, 63
row, 57
rownames(), 63
selecting columns, 59
selecting elements, 59
selecting rows, 59
sorting, 65
subset(), 60
tail(), 75
with(), 69
data.frame(), 62
data.matrix(), 178
dataEllipse(), 255
dbinom( ), 114, 141
dchisq( ), 131
decisions, 184
degree of freedom, 130
degrees of freedom, 131, 132
density( ), 150
dependent events, *see* probability
descriptive statistics, 15
deviation, 31
dichotomous data, 16
discrete data, 16
distorted graph, 18
dnorm( ), 127
dpois( ), 119
draws, 149–151
compute probabilities with, 151
extract( ), 150
sample, 190
subtracting draws, 159
dummy variables, 275

ecdf(), 151
effect size, 229
error, 256
expression(), 163, 169, 181, 182, 233
extract( ), 150

F density, 132
definition, 132
degrees of freedom, 132
F-test, 217
pf( ), 132
use in ANOVA, 240
F-test, *see* NHST
factor, 51
factor(), 51
levels(), 51
factorial( ), 108
FCI, *see* confidence interval
fences, 41
five number summary, 40
for(), 73, 182
frequency, 17
frequentist, *see* probability
frequentist confidence interval, *see* confidence interval

gaussian, *see* normal density
generalized linear model, 293
glm( ), 293
link function, 293
GLM, *see* generalized linear model
glm( ), 293
graphs
bar plot, 76
boxplot, 41, 75
correlogram, 265
heat map, 254
histogram, 17
pairs(), 70, 79
pie chart, 18
Q-Q plot, 255
scatter plot, 18, 68
violin plot, 159, 181
grid(), 68, 115

hdi(), 171
head(), 75
heat map, 254
highest density interval, 171
hist( ), 17
histogram, 17
hypothesis, 137
alternate hypothesis in NHST, 229
NHST, 209
null hypothesis in NHST, 209

hypothesis testing, *see* NHST

`I()`, 280
independent events, *see* probability, 93
index, 20
indifference principle, *see* principle of insufficient reason
inferential statistics, 15, 81
`install.packages()`, 159
installing R packages, 159
interquartile range, 40, 46
IQR, *see* interquartile range

`jitter()`, 71, 255
joint posterior density, 167

`lapply()`, 179, 234, 268
`legend()`, 68
`length()`, 62
`length( )`, 151, 159
levels, *see* factor
likelihood, 99, 139
`lines()`, 158
`lines( )`, 18
list, 62
`list()`, 72, 179
`lm( )`, 260
logarithmic transformation, 283
logistic regression, *see* regression
logit, 289
loops, 73, 182
lurking variable, 266

making decisions, 184
marginal posterior density, 168
masking, 265
matrix, 178
MCMC, 149
mean, 27
  `mean( )`, 29
`mean( )`, 29
measurements, *see* data
median, 26
  `median()`, 26
`median`, 26
minimum detectable difference, 232
missing values, 54
mode, 25
multi-linear regression, *see* regression
mutually exclusive, 94

`NA`, *see* missing values
`NaN`, 54
NHST
  alternate hypothesis, 229
  ANOVA, 239
    `aov( )`, 242
    between groups SS, 240
    error SS, 240
    vs Bayesian analysis, 244
  compound hypothesis, 209
  density of p under H_0, 221
  F-test, 217
  flaws, 220
  null hypothesis, 209
  one-tailed test, 212
  p-value, 212, 221
  p-value fishing, 224
  p-value lottery, 221
  power, 229
    required sample size, 234
  significance level, 212
  statistical power, 230
  statistically significant, 212
  t test, 210
  two-tailed test, 212
  Type-I error, 229
  Type-II error, 229
  variance tests, 217
n-1, *see* Bessel's correction
noise, 9, 75, 249
nominal data, 16
normal density, 127
  cumulative function, 127
  `dnorm( )`, 127
  `pnorm( )`, 127, 151
  `qnorm( )`, 202
  `rnorm( )`, 150
  standard normal density, 128, 193
  standard score, 128
  z density, 128, 193
`NULL`, 54
null hypothesis, 209
null hypothesis significance testing, *see* NHST

odds, 289
`optimizing()`, 181
order statistic, 38
`order()`, 65
ordinal data, 16
outlier, 41
outliers, 46
overfitting, 269, 282

p-value, 212, 221
  density under H_0, 221
  fishing, 224
  lottery, 221

p-value fishing, 224
p-value lottery, 221
paired data, 18
`pairs()`, 70, 79
`par()`, 81
partial residual plot, 271
`paste()`, 163, 169, 233
`pbinom()`, 116
`pchisq( )`, 131
Pearson's correlation coefficient
  `cor( )`, 266
percentile, *see* qantile38
`pf( )`, 132, 240
pie chart, 18
`plot( )`, 18, 115
`pnorm( )`, 127, 151
point estimate, 8, 138, 143
Poisson Distribution, 118, 164
  `dpois( )`, 119
pooled standard deviation, 230
population, 190
posterior density, 143
  draws, 149–151
  `extract( )`, 150
  joint, 168
  marginal, 168
posterior probability, 99, 139
power, *see* NHST
predicted value, 270
prediction interval, 273
predictor variable, 249
principle of insufficient reason, 89
`print()`, 40
prior probability, 99, 139
probability, 85
  add up to one, 97
  as plausibility, 85
  Bayes' Theorem, 99, 139
  binomial, 107
  complement rule, 89
  conditional, 86
  cumulative density
    `ecdf()`, 151
    from draws, 151
    `pchisq( )`, 131
    `pf( )`, 132, 240
    `pnorm( )`, 127, 151
    `pt( )`, 132, 211
  cumulative distribution, 116
  density function, 125
    Behrens-Fisher, 206
    beta, 153
    chi-squared, 130, 194
    χ^2, 130, 194
    `dchisq( )`, 131
    `density( )`, 150
    `dnorm( )`, 127
    F density, 132, 217
    normal, 127
    posterior density, 143
    standard normal, 128
    t, 131
    uniform, 140
    Welch's t, 206
    z density, 128
  dependent events, 92
  distribution function, 114
    binomial, 114
    `dbinom( )`, 114, 141
    `dpois( )`, 119
    Poisson, 118, 164
  frequentist intepretation, 189
  independent events, 92, 93
  likelihood, 99, 139
  mutually exclusive, 94
  posterior probability, 99, 139
  principle of insufficient reason, 89
  prior probability, 99, 139
  product rule, 91
  standard score, 128
  subjectivity of, 86
  sum rule, 93
probability density, 125
probability distribution, *see* probability
product rule, 91
`pt( )`, 132, 211

Q-Q Plot, 255
`qnorm( )`, 202
qualitative data, 15
quantile, 37, 128, 149
  interquartile range, 40
  order statistic, 38
  percentile, 38
  `qnorm( )`, 202
  `quantile( )`, 37, 38
  quartile, 38
`quantile( )`, 40, 150
quantitative data, 15
quartile, *see* qantile38, 40

r, *see* Pearson's correlation coefficent
R^2, *see* coeffecient of determination
range, 40
`rbind()`, 64
`read.csv()`, 64
regression, 7, 249
  adjusted R^2, 269

Bayesian parameter fitting, 249
coefficient of determination, 258
definition, 249
generalized linear model, 289, 293
`glm( )`, 293
interaction, 286
least-squares fitting, 249, 260
linear, 249
link function, 293
`lm( )`, 260
logistic, 289
main effects, 286
multi-linear, 263
multilinear, 263
multivariable, 263
nonlinear, 279
overfitting, 269, 282
parameters, 249
partial residual plot, 271
polynomial, 280
predicted value, 270
prediction interval, 273
predictor variable, 249
R^2, 258
response variable, 249
`stan_glm()`, 252, 291
`visreg()`, 260, 271, 281
`rep()`, 242
`replicate()`, 192
residual, 28, 32, 256
response variable, 249
row, 17, 57
`rownames()`, 63
`rstan`, 147, 149

s, *see* standard deviation
sample, 190
density of sample means, 191
density of sample variances, 194
size required in NHST, 234
sample mean distribution, 190
for non-normal data, 195
scatter plot, 18
`sd( )`, 35
se, 196
sem, 196
sensitivity analysis, 172
`seq()`, 40
`seq( )`, 142
sigma notation, 19
significance level, 212
simulated data, 150
SMD, *see* sample mean distribution
`smoothScatter()`, 255
`sort( )`, 17, 44
s^2, *see* variance
Stan, 147
compute probabilities w/ draws, 151
defining a model, 147
`extract( )`, 150
`Stan( )`, 149
`Stan( )`, 149
`stan_glm()`, 252, 291
standard deviation, 35
pooled, 230
`sd( )`, 35
standard error of the mean, 196
standard normal density, *see* normal density
standard score, *see* probability
standardized residuals, 255
statistical hypothesis, 137
statistical inference, *see* inferential statistics
statistical model, 139
statistical power, 230
statistically significant, 212
student's t density, *see* t density
subscript notation, 19
`subset()`, 44, 60
sufficient statistic, 138
sum of squares, 240
sum rule, 93
`sum( )`, 19, 142, 143
`summary()`, 40, 51
summation notation, 19

t density, 131
definition, 131
degrees of freedom, 131
density of sample means, 194
`pt( )`, 132, 211
t-test, 210
`t.test()`, 203, 213
table, 17
`tail()`, 75
`text()`, 255
time series, 18
Type-I error, 229
Type-II error, 229

unbiased estimator, 34
`unclass()`, 52, 68
uniform density, 140
`unname()`, 46

`var( )`, 34
variance, 34

Bessel's correction, 34
`var( )`, 34
vector, 17
`c( )`, 17
creating, 17
factor, 51
`factor()`, 51
selecting elements, 43
`sort()`, 44
`subset()`, 44
`summary()`, 40, 51
violin plot, 159, 181
`vioplot()`, 159, 181
`vioplot()`, 159, 181
`visreg()`, 260, 271, 281

Welch's t density, 206
`with()`, 69, 281
`write.csv()`, 64

z density, 128, 193